Úkol č. 3: Množinové operácie s polygónmi

Hugo Majer, Júlia Šušková

22. mája 2022

1 Zadanie

Úloha č. 4: Množinové operace s polygony

Vstup: nekonvexní polygony P, Q,

Výstup: množina k polygonů $P' = \{P'_1, ..., P'_k\}$.

S využitím algoritmu pro množinové operace s polygony implementujte pro dvojici polygonů P, Q následující množinové operace:

- průnik polygonů,
- sjednocení polygonů,
- rozdíl polygonů.

Jako vstupní data použijte existující kartografická či syntetická data reprezentující množiny P, Q, která budou načítána ze dvou textových souborů ve Vámi zvoleném formátu.

Grafické rozhraní realizujte s využitím frameworku QT, výsledky množinových operací vizualizujte.

2 Údaje o bonusových úlohách

Neboli riešené žiadne bonusové úlohy.

3 Množinové operácie polygónov

Základné množinové operácie – zjednotenie, prienik, rozdiel majú veľký význam nielen v matematike, ale aj v oblasti GIS a v kartografii. Na vstupe máme dve množiny, ktoré predstavujú uzavreté, ohraničené oblasti (polygóny) A a B.

Zjednotenie $A \cup B$ je taká množina, ktorá obsahuje všetky prvky, ktoré sa nachádzajú aspoň v jednej z množín, ktoré zjednocujeme a žiadne ďalšie prvky. Je to asociatívna a komutatívna operácia.

Prienik $A \cap B$ je taká množina, ktorá obsahuje práve tie prvky množiny A, ktoré sú súčasne prvkami množiny B. Je to asociatívna a komutatívna operácia.

Rozdiel $A \cap \overline{B}$, resp. $B \cap \overline{A}$ je taká množina, ktorá obsahuje každý prvok, ktorý se nachádza v prvej množine, ale nenachádza se v druhej a žiadne ďalšie prvky. Nie je to ani asociatívna ani komutatívna operácia.

3.1 Problémové situácie

V prípade vstupu dvojice polygónov do množinovej operácie je prvou singularitou, ktorá sa môže vyskytnúť, kolinearita hrán, tj. spoločná hrana vstupných polygónov, alebo len jej časť. Výsledkom zjednotenia by táto kolineárna hrana byť nemala. V prípade prieniku by výsledkom bola práve táto kolineárna hrana. Pri rozdiele je výsledkom jeden zo vstupných polygónov.

Môže nastať aj situácia, kedy polygóny majú spoločný vrchol, resp. vrcholy, alebo vrchol polygónu leží na hrane druhého polygónu. V prípade zjednotenia by výsledkom mali byť oba polygóny. Pri prieniku by výsledkom mal byť práve ten spoločný vrchol, resp. vrcholy, čiže bod(y).

Spomínané singularity **neboli** v rámci tejto práce riešené a ošetrené.

4 Popis algoritmu

Na vstupe do algoritmu máme dva polygóny $A = \{P_i\}_{i=1}^{n}$ a $B = \{Q_j\}_{j=1}^{m}$, ktorých prvý a posledný bod je totožný a majú CCW orientáciu. Základom algoritmu je nový dátový typ s názvom `QPointFB`, ktorý obsahuje informáciu o parametroch α, β, ktoré popisujú polohu priesečníku dvoch hrán v rámci týchto dvoch hrán. Takisto nesie informáciu o polohe bodu voči polygónu.

Samotný algoritmus pozostáva z viacerých častí.

4.1 Výpočet priesečníkov

Pozostáva z prechádzania všetkých hrán polygónov A, B a počíta sa ich priesečník b_{ij}, ak existuje. Najprv je nutné poznať vzájomnú pozíciu, resp. vzťah týchto dvoch hrán, nakoľko nás zaujímajú len tie, ktoré sa pretínajú. Máme dve hrany $e(P_i, P_{i+1})$, $f(Q_j, Q_{j+1})$. Stanovme, že $P_i = [x_i, y_i]$, $P_{i+1} = [x_{i+1}, y_{i+1}]$, $Q_j = [x_j, y_j]$, $Q_{j+1} = [x_{j+1}, y_{j+1}]$. Potrebujeme spočítať smerové vektory $\vec{u}$, $\vec{v}$ a $\vec{w}$

$$\vec{u} = (x_{i+1} - x_i, y_{i+1} - y_i),$$
$$\vec{v} = (x_{j+1} - x_j, y_{j+1} - y_j),$$
$$\vec{w} = (x_i - x_j, y_i - y_j).$$

Následne zo zložiek vektorov spočítame determinanty k_1, k_2, k_3

$$k_1 = \begin{vmatrix} v_x & v_y \\ w_x & w_y \end{vmatrix} = v_x * w_y - v_y * w_x,$$
$$k_2 = \begin{vmatrix} u_x & u_y \\ w_x & w_y \end{vmatrix} = u_x * w_y - v_y * w_x,$$
$$k_3 = \begin{vmatrix} u_x & u_y \\ v_x & v_y \end{vmatrix} = u_x * v_y - u_y * v_x.$$

Z determinantov spočítame parametre α, β

$$\alpha = \frac{k_1}{k_3}, \qquad \beta = \frac{k_2}{k_3}.$$

Ak $k_1 = 0$, $k_2 = 0$, $k_3 = 0$, tak hrany e, f sú kolineárne, priesečník neexistuje.
Ak $(k_1 = 0 \wedge k_2 = 0)$, tak hrany e, f sú rovnobežné, priesečník neexistuje.
Priesečník existuje ak $(0 \leq \alpha \leq 1) \wedge (0 \leq \beta \leq 1)$. Následne je nutné spočítať jeho súradnice x, y

$$x = x_i + \alpha u_x,$$
$$y = y_i + \alpha u_y.$$

V každom inom prípade sú hrany e, f mimobežné.

4.2 Vloženie priesečníkov do polygónov

V predošlom kroku môžu nastať situácie, kedy hrana f polygónu B pretína hranu e polygónu A najviac jedenkrát alebo situácia, kedy hrana e polygónu A môže byť pretnutá viacerými hranami f_i polygónu B. Je teda nutné testovať hranu polygónu A voči všetkým hranám polygónu B.

Ak existuje priesečník b_{ij}, pre polygón B bude spracovaný priamo, tj. pridaný do polygónu B na pozíciu $i+1$, tj. medzi body Q_i a Q_{i+1}. Pre polygón A nie je ešte možné priesečník b_{ij} pridať medzi P_i a P_{i+1}. Priesečník musí byť pridaný do pomocnej štruktúry – slovníku $D(\alpha) = (\alpha_i, b_{ij})$, pretože ešte nepoznáme všetky priesečníky. Po nájdení všetkých priesečníkov sa hromadne aktualizuje polygón A, tj. slovník $D(\alpha)$ pridaný medzi body P_i a P_{i+1}.

Doposiaľ vysvetlený postup algoritmu pseudokódom zapíšeme nasledovne:

Algorithm 1 *Nájdenie priesečníkov a vloženie priesečníkov do polygónov (A, B)*

Inicializuj $i = 0$.
Pokým $i <$ length(A):
    $D(\alpha) = \{\}$ # Vytvorenie slovníku
    Inicializuj $j = 0$.
    Pokým $j <$ length(B):
        Nájdi priesečník b_{ij}. # Detailne popísané v kap. 4.1
        Ak $\exists b_{ij} = (P_i, P_{i+1}) \cap (Q_j, Q_{j+1})$:
            $D(\alpha_i) \leftarrow b_{ij}$. # Pridaj priesečník do D
            j += 1 # Inkrementuj pozíciu v rámci B
            $B \leftarrow (j, b_{ij})$ # Vlož priesečník na pozíciu $j+1$ do B
        j += 1.
    Ak $D(\alpha) \neq \varnothing$: # Bol nájdený priesečník
        Pre $\forall$ (key, $value$) $\in D(\alpha)$: # Prechádzaj všetky prvky (priesečníky) v D
            i += 1. # Inkrementuj pozíciu v rámci A
            $b_{ij} \leftarrow value$ # Druhá hodnota z páru v D je priesečník
            $A \leftarrow (i, b_{ij})$ # Vlož priesečník na pozíciu $i+1$ do A
    i += 1.

4.3 Ohodnotenie vrcholov polygónov

Spočíva v určení polohy hrany jedného polygónu voči druhému polygónu. Prvým krokom je spočítanie stredového bodu $\overline{P}$ hrany $e(P_i, P_{i+1})$ (patriacej polygónu A) a analogicky bodu $\overline{Q}$ pre hranu $f(Q_i, Q_{i+1})$. Každému $\overline{P} \in A$ určujeme polohu voči polygónu B, každému $\overline{Q} \in B$ voči polygónu A. Jednotlivé hrany polygónov tým pádom budú rozdelené na vnútorné a vonkajšie. Táto informácia je uložená do počiatočného bodu každej hrany. Ak by sme chceli riešiť singularity uvedené v kapitole 3.1, je nutné rozlíšiť aj stav na hrane polygónu. Nakoľko sa práca singularitám nevenuje, v samotne implementácií algoritmu si postačíme len s rozlíšením hrán na vnútorné/vonkajšie.

$$\text{pos}\,(e_i, B) = \text{pos}\,(\overline{P_i}, B) = \begin{cases} \overline{P_i} \in B, \\ \overline{P_i} \in \partial B, \\ \overline{P_i} \notin B, \end{cases} \qquad \text{pos}\,(f_i, A) = \text{pos}\,(\overline{Q_i}, A) = \begin{cases} \overline{Q_i} \in A, \\ \overline{Q_i} \in \partial A, \\ \overline{Q_i} \notin A. \end{cases}$$

S požiadavkou stanovenia polohy stredového bodu hrany voči polygónu sa vraciame ku starému známemu *Point in Polygon Test*.

Poznámka: V rámci tejto práce bol využitý *Winding Number Algorithm*, ktorý bol detailne popísaný v predošlom úkole a nie je popisovaný znova.

Algorithm 2 *Ohodnotenie vrcholov polygónov (A, B)*

1: Pre $\forall$ hrany $e_i(P_i, P_{i+1})$ polygónu A:
2: Nájdi stredový bod $\overline{P}$: $\overline{P} = 0,5(P_i + P_{i+1})$.
3: Zisti pozíciu $\overline{P}$ voči polygónu B.
4: Nastav zistenú pozíciu počiatočnému bodu hrany $e_i(P_i, P_{i+1})$.
5: Pre $\forall$ hrany $f_i(Q_i, Q_{i+1})$ polygónu B:
6: Nájdi stredový bod $\overline{Q}$: $\overline{Q} = 0,5(Q_i + Q_{i+1})$.
7: Zisti pozíciu $\overline{Q}$ voči polygónu A.
8: Nastav zistenú pozíciu počiatočnému bodu hrany $f_i(Q_i, Q_{i+1})$.

4.4 Výber hrán podľa pozície

Dochádza k výberu hrán z oboch polygónov, ktoré majú voči druhému polygónu určitú pozíciu, ktorá bola určená v predošlom kroku. Zadaná pozícia je daná typom množinovej operácie, ktorú chceme vykonať. V rámci tohto kroku dochádza ku prechádzaniu cez všetky vrcholy polygónu (cez vrcholy, pretože pozícia hrany bola ukladaná do počiatočného bodu hrany), ak sa pozícia bodu rovná zadanej pozícií, tak z tohto bodu a z nasledujúceho je vytvorená hrana. Tieto hrany budú výsledkom, tj. budú to hrany výsledného polygónu po aplikácií danej množinovej operácie.

V prípade operácie zjednotenia sú z polygónu A, resp. polygónu B vybrané hrany, ktoré sú mimo polygónu A, resp. polygónu B. V prípade operácie prieniku sú z polygónu A, resp. polygónu B vybrané hrany, ktoré sú vnútri polygónu A, resp. polygónu B. V prípade operácie rozdielu A-B sú z polygónu A vyberané hrany mimo polygónu B a z polygónu B hrany, ktoré sú vo vnútri polygónu A. Presne opačne to platí pre operáciu B-A.

Algorithm 3 *Výber hrán podľa pozície voči A/B z polygónu A/B*

1: Pre $\forall$ vrcholy P_i polygónu A:
2: Ak jeho pozícia $\equiv$ zadaná pozícia: # Zadaná pozícia podľa požadovanej operácie
3: Vytvor hranu (P_i, P_{i+1}).
4: Pre $\forall$ vrcholy Q_i polygónu B:
5: Ak jeho pozícia $\equiv$ zadaná pozícia: # Zadaná pozícia podľa požadovanej operácie
6: Vytvor hranu (Q_i, Q_{i+1}).

5 Aplikácia

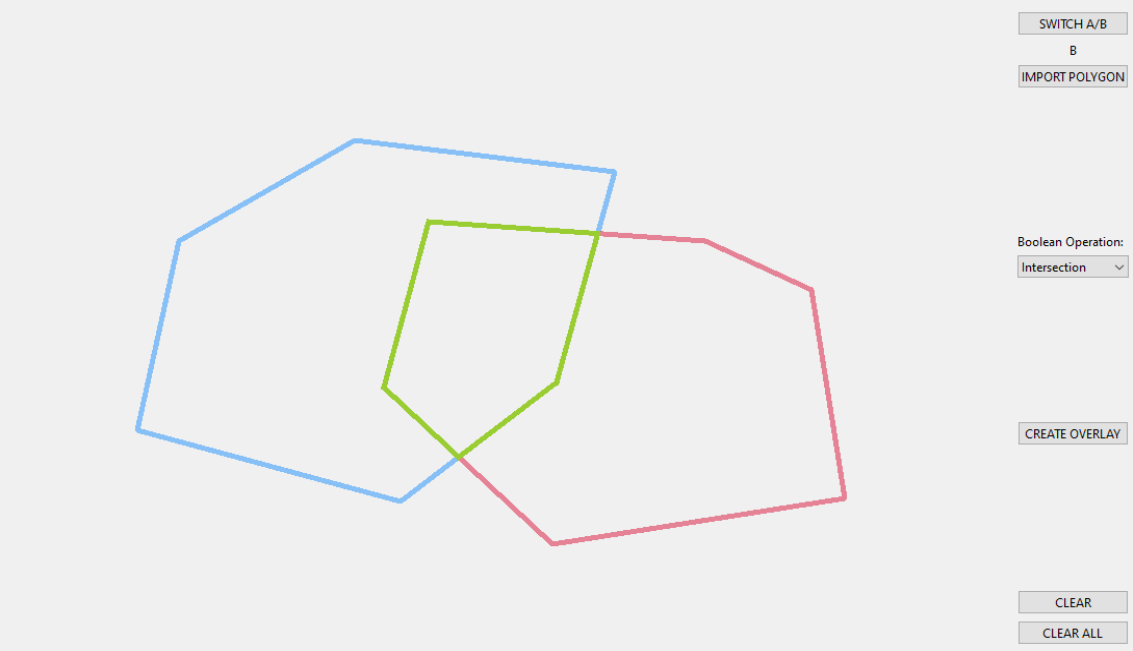
Aplikácia bola vytvorená vo vývojovom prostredí QT 6. Slúži na vykonanie množinových operácií zjednotenie, prienik, rozdiel dvoch uzavretých, ohraničených oblastí (polygónov). Užívateľské rozhranie je veľmi jednoduché a intuitívne. Prevažnú časť okna aplikácie tvorí zobrazovacia plocha, na ktorej sa vizualizujú do aplikácie nahrané dáta a výsledky operácií.

Vstupné dáta užívateľ do aplikácie nahrá z textového súboru obsahujúceho súradnice X, Y vrcholov jedného polygónu. Súradnice musia byť v súradnicovom systéme zobrazovacej plochy. Súradnice vrcholov druhého polygónu sú importované z druhého textového súboru. Na rozlíšenie, ktorý polygón je práve z textového súboru importovaný, je použité tlačidlo *SWITCH A/B*. Pod týmto tlačidlom sa nachádza aj indikátor, ktorý polygón je práve zvolený a ktorý bude teda importovaný.

Druhou možnosťou je polygóny nakresliť a to priamočiaro – klikaním na zobrazovaciu plochu. Opäť po nakreslení prvého polygónu je nutné kliknúť na tlačidlo *SWITCH A/B* a nakresliť druhý.

Následne užívateľ vyberie z combo-boxu požadovaný typ množinovej operácie a po kliknutí na tlačidlo *CREATE OVERLAY* sa operácia vykoná a zobrazí sa výsledok (zelenou farbou).

Tlačidlo *CLEAR* vráti zobrazovaciu plochu do stavu pred vykonaním množinovej operácie, tj. zostanú na nej len vstupné polygóny. Tlačidlo *CLEAR ALL* okrem výsledku operácie zmaže aj vstupné polygóny.



Obr. 1: Rozhranie aplikácie s polygónmi a vykonanou operáciou.

6 Dokumentácia

1. Trieda `Mainform`:

   Metóda `clickImport` inicializuje vloženie vstupných dát do programu. Metóda volá funkciu `setPath`.

   Metóda `clickSwitch` slúži pre zmenu importovaného/kresleného polygonu z polygonu A na polygon B.

   Metóda `clickCreateOverlay` inicializuje výpočet množinových operácií a vizualizáciu výsledku.

   Metóda `clickClear` slúži na zmazanie výsledku konkrétnej množinovej operácie.

   Metóda `clickClearAll` slúži na zmazanie výsledkov ako aj vložených dát.

2. Trieda `Draw`: Je zodpovedná za grafickú stránku aplikácie a zobrazovacej plochy. Trieda obsahuje inicializátor a 8 metód:

   `Inicializátor` má dva pozičné argumenty a inicializuje premenné pre túto triedu. Inicializované sú prázdne listy pre vstupné body polygonov A a B, list pre výsledky množinových operácií a premennú slúžiacu k aktivácii polygonu B z polygonu A.

   Metóda `switchPolygon` slúži k aktivácii zmeny z polygonu A na polygon B.

   Metóda `getPolygons` vracia konkrétne premenné triedy `Draw`.

   Metóda `setResults` vyžaduje jeden argument, ktorý je následne metódou priradený konkrétnej premennej triedy `Draw`.

   Metóda `clearResults` slúźi k zmazaniu dát v premennej `res`.

   Metóda `clearCanvas` slúži k zmazaniu dát v premenných `res`, `polA`, `polB`.

   Metóda `setPath` slúži k načítaniu vstupných dát. Do premenných `polA` a `polB` triedy ukladá vstupné body načítané z `TXT` súboru. Na prečitanie súboru sa používa externý modul `csv`.

   Metóda `mousePressEvent` slúži užívateľovi k manuálnemu výberu polygonov pomocou myši priamo na zobrazovacej ploche aplikácie.

   Metóda `paintEvent` slúži k vizualizácii polygonov a výsledkov množinových operácií.

3. Trieda `QPointFB`: Táto trieda je odvodená z rodičovskej triedy `QPointF`. Obsahuje `Inicializátor` a 4 metódy:

   `Inicializátor` má 5 argumentov, z toho 4 dátového typu `float` a 1 `enum` odvodenej triedy

PointAndPolygonPosition. Argumenty `x, y` predstavujú priestorové súradnice bodu. Argumenty `alpha` a `beta` defaultne nulové. Argument `pos` je takisto nastaveý defaultne.

Metódy `getAlpha`, `getBeta` a `getPosition` vracajú hodnoty premenných `aplha, beta, pos`.

Metódy `setPosition` má jeden argument, ktorý je v metóde priradený premennej `pos`.

4. Trieda `Edge`: Trieda definuje líniu tvorenú počiatočným a koncovým bodom typu `QPointFB`. Obsahuje `Inicializátor` a 2 metódy:

   `Inicializátor` má 2 argumenty typu `QPointFB`, ktoré predstavujú počiatočný a koncový bod línie.

   Metódy `getStart` a `getEnd` vracajú počiatočný alebo koncový bod línie.

5. Trieda `BooleanOperation`: Trieda je odvodená z triedy `Enum` a slúži k priradeniu symbolických názvov ku konštantným hodnotám pre 4 množinové operácie - `Union, Intersection, DifferenceAB, DifferenceBA`.

6. Trieda `PointAndLinePosition`: Trieda je odvodená z triedy `Enum` a slúži k priradeniu symbolických názvov ku konštantným hodnotám pre 3 možné pozície bodu voči úsečke - `LeftHP, RightHP, OnLine`.

7. Trieda `LineAndLinePosition`: Trieda je odvodená z triedy `Enum` a slúži k priradeniu symbolických názvov ku konštantným hodnotám pre 3 možné pozície dvoch úsečiek - `Parallel, Skew, Collinear, Intersect`.

8. Trieda `PointAndPolygonPosition`: Trieda je odvodená z triedy `Enum` a slúži k priradeniu symbolických názvov ku konštantným hodnotám pre 3 možné pozície bodu voči polygonu - `Inside, Outside, OnBoundary`.

9. Trieda `Algorithms`: Trieda obsahuje matematické metódy, pomocou ktorých sú realizované jednotlivé algoritmy, ktoré sú taktiež zapísané v tejto triede.

   Metóda `getPointAndLinePosition` má na vstupe tri argumenty, všetky predstavujú body v dátovom type `QPointFB`. Slúži na určenie polohy analyzovaného bodu a priamky. V prvom kroku je definovaná prijateľná odchýlka epsilon, nasleduje výpočet zložiek dvoch vektorov a výpočet vektorového súčinu pomocou determinantu a testovanie podmienok, do ktorých vstupuje hodnota determinantu a podľa ktorej sa určí pozícia bodu voči priamke. Metóda vracia hodnotu 1 ak je bod v ľavej polrovine, hodnotu 0 ak je bod v pravej polrovine, v prípade kolinearity vráti hodnotu -1.

   Metóda `get2LinesAngle` má na vstupe štyri body, opäť všetky v dátovom type `QPointFB`.

Slúži na výpočet uhlu dvoch priamok. Opäť sú spočítané dva vektory, ich skalárny súčin a ich norma. Metóda vracia absolútnu hodnotu skalárneho súčinu.

Metóda `getPositionPointAndPolygon` má na vstupne jeden bod dátového typu `QpointFB` a polygon v tvare listu `QpointFB`. Metóda analyzuje pozíciu bodu voči polygonu - vnútri polygonu, mimo polygonu alebo na hrane polygonu - a vracia hodnotu v dátovom tvare triedy `PointAndPolygonPosition`.

Metóda `get2LinesIntersection` má na vstupe 4 body v tvare `QPointFB` a slúži k výpočtu súradníc priesečníku dvoch úsečiek tvorených vstupnými argumentami. Metóda vracia pozíciu týchto dvoch úsečiek voči sebe v dátovom tvare triedy `LineAndLinePosition` a v prípade, ak existuje priesečnik, tak tento je vracaný v tvare `QPointFB`.

Metóda `updateVertices` má na vstupe dva polygonu A a B v tvare listu `QPointFB`. Slúži k priradeniu nových vrcholov ku aktivovanému polygónu, ktoré vznikli ako priesečníky strán polygonu A a B. metóda nevracia žiadnu hodnotu, avšak aktualizuje už existujúce listy vrcholov vstupných polygonov a zoraďuje ich podľa hodnôt alpha/beta v narastajúcom poradí.

Metóda `setEdgePosition` má na vstupe opäť dva polygony A a B v tvare listu `QPointFB`. Metóda slúži k uloženiu pozície hrán aktivovaného polygonu voči neaktivovanému polygonu. Pre všetky hrany polygonu je vypočítaný ich stred. Následne sa analyzuje a uloží pozícia tohto stredu voči polygonu B a je uložená pre vrcholy polygonu A v premennej `pos`.

Metóda `getEdges` má na vstupe tri argumenty - polygon v tvare listu `QPointFB`, indikátor pozície v tvare triedy `PointAndPolygonPosition` a zoznam hrán triedy `Edge`. Metóda priraďuje zoznamu `edge` hrany, ktorých vrcholy majú určitú pozíciu.

Metóda `createOverlay` má na vstupe dva polygony v tvare listu `QPointFB` a indikátor množinovej operácie. Výstup je zoznam hrán, ktoré sú výsledkom určitej množinovej operácie nad vstupnými polygonmi.

7 Záver

Vytvorená aplikácia úspešne vykonáva množinové operácie zjednotenia, prieniku a rozdielu dvoch polygónov, ktoré užívateľ importuje z dvoch textových súborov so súradnicami vrcholov polygónov, alebo ich nakreslením na zobrazovaciu plochu aplikácie.

Jediný nedostatok aplikácie sa vyskytuje pri operácií zjednotenia, ktorá nie vždy dáva úplne správne výsledky. Z neznámeho dôvodu je v takomto prípade nutné ešte raz (znovu) kliknúť na *CREATE OVERLAY* a výsledok by už mal byť správny.

Námetom na vylepšenie môže byť ošetrenie singularít (viz kapitola 3.1), ktorých riešenie je bonusovou úlohou, ktorá riešená nebola.

8 Použité zdroje

Prednášky a cvičenia z predmetu *Algoritmy počítačové kartografie*.