

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA STAVEBNÍ, OBOR GEODÉZIE A KARTOGRAFIE
KATEDRA GEOMATIKY

název předmětu

GEOINFORMATIKA

| | | | | |
|------------------------------|--|---|------------------------------|--------------------|
| <i>číslo úlohy</i> 2 | <i>název úlohy</i> Vyhledávání objektů v mapách | | | |
| <i>školní rok</i> 2025/26 | <i>skupina</i> 2 | <i>zpracovali</i> Králič Adam, Matějková Barbora | <i>datum</i> 18. 11. 2025 | <i>klasifikace</i> |

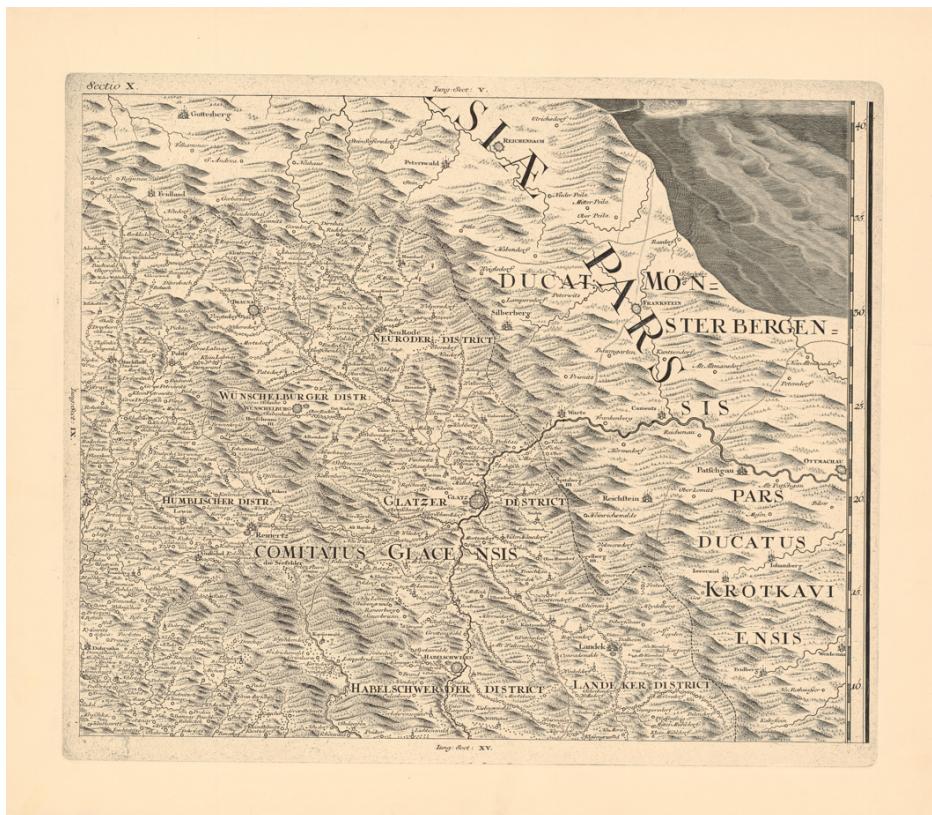
Technická zpráva

1 Zadání úlohy

Úkolem bylo aplikovat algoritmy pro vyhledávání objektů/ploch v rastrových datech.

1.1 Část A

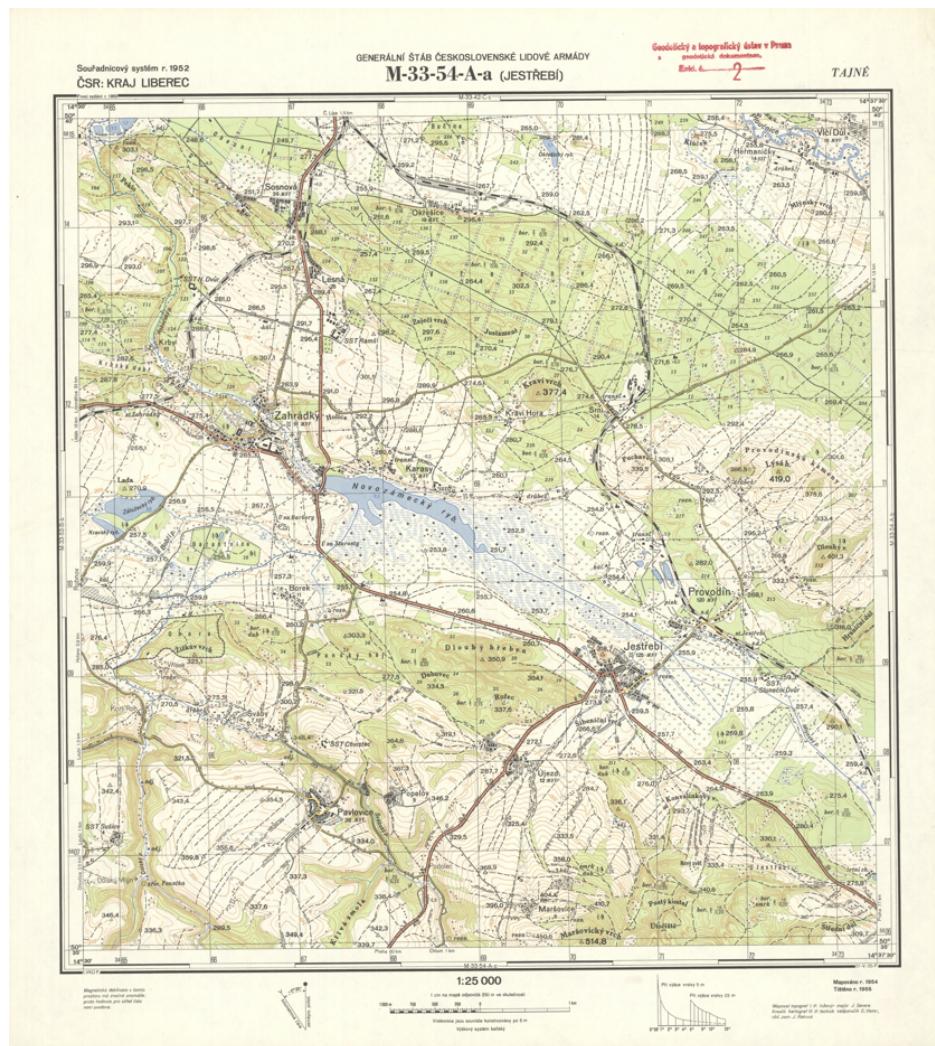
Implementovat algoritmus pro vyhledávání vzorů v Müllerově mapě Čech (obr. 1) na základě obrazové korelace. Vybrat jeden z objektů typu obec s kostelem jako okno pro vyhledávání (bonus: průměr z 5 vzorů) a na základě vhodné hodnoty korelace vyhledat všechny pozice obcí s kostelem na mapovém listu. Výsledek odevzdat jako pixelové souřadnice kostelů (bonus: proředit o vícenásobné opakování).



Obrázek 1: zadaná část Müllerovy mapy Čech

1.2 Část B

Implementovat algoritmus obrazové segmentace pomocí metody **k-Means** nad obrazem Topografické mapy ČSSR v měřítku 1 : 25 000 (obr. 2). Výsledkem bude extrahovaná plocha lesů očištěná o vrstevnice a další překryvnou kresbu, naopak budou z lesů odečteny lesní průseky. Zachovat otvory v lesním prostoru větší než 50 pixelů a výsledek odevzdat jako pole pixelových souřadnic.



Obrázek 2: Topografická mapa ČSSR v 1 : 25 000; M-33-54-A-a (Jestřebí)

| Krok | Hodnocení |
|--|-------------|
| Vyhledané pixelové souřadnice obcí s kostelem na mapovém listu. | 20 b |
| Souřadnice prověřené o vícenásobná opakování. | +5 b |
| Skript přijímající jako uživatelský vstup pět vzorů obcí s kostelem, které jsou průměrovány. | +10 b |
| Plocha lesa očištěná o zbytkovou kresbu | 20 b |
| Segmentovaný obraz mapy při použití Gaborova nebo STD filtru | +10 b |
| Plocha lesa uložená jako vrstva v GeoPackage v systému UTM | +10 b |
| Technická zpráva. | 10 b |
| Max celkem: | 85 b |

Tabulka 1: Povinné a bonusové úlohy

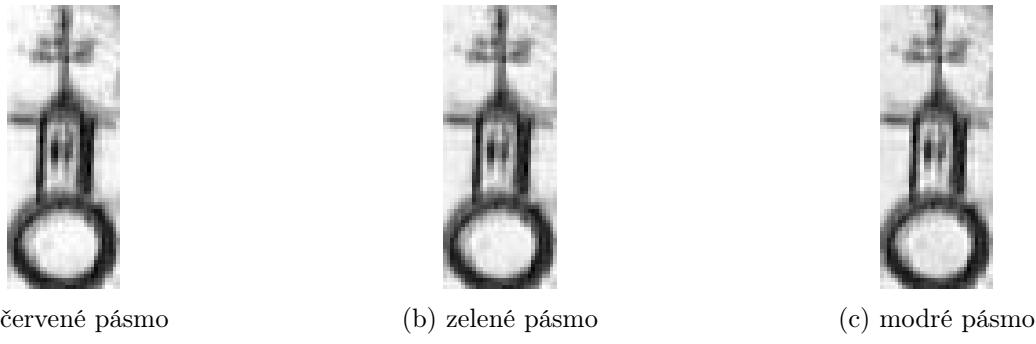
2 Postup výpočtu

2.1 Část A

Veškeré výpočty proběhly za využití programovacího jazyka Python.

Nejprve byla Müllerova mapa Čech ve formátu jpg načtena do Numpy pole pomocí knihovny `rasterio`, bylo zvoleno jedno ze 3 pásem RGB; pro veškeré následující výpočty bylo použito pásmo červené a hodnoty jednotlivých pixelů byly převedeny z formátu `int` v rozsahu $\{0, 1, \dots, 255\}$ na `float` z intervalu $[0, 1]$ pro efektivnější výpočty.

Následně byly v ilustračním programu Krita odečteny pixelové souřadnice hledaného objektu na 5 různých místech mapy, šířka a výška vzoru byla stanovena jako nejmenší ohraničující obdélník objektu, čímž byla určena i velikost vyhledávacího okna. Těchto pět nalezených objektů bylo zprůměrováno do hledaného vzoru (obr. 3).



Obrázek 3: průměrované vzory podle jednotlivých pásem

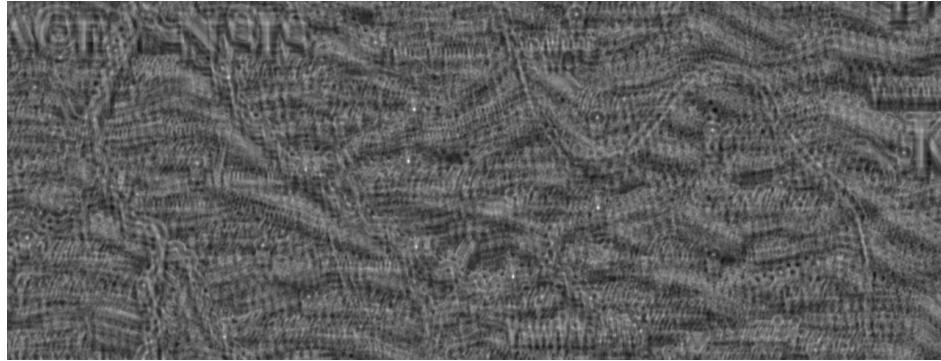
Při použití různých barevných pásem se hodnoty tohoto vzoru příliš nelišíly, nejvíce se z této skupiny vymyká vzor při použití modrého pásma.

Souřadnice hledaných objektů byly lokalizovány pomocí normalizovaného vzájemného korelačního koeficientu (rov. 1, kde A je vzorová kresba kostela, B potom obsah vyhledávacího okna).

$$\begin{aligned}
C &= \frac{1}{I \cdot J} \sum_{j=1}^J \sum_{i=1}^I \frac{(A(i,j) - \bar{A}) \cdot (B(i,j) - \bar{B})}{\sigma_A \cdot \sigma_B} \\
\bar{A} &= \frac{1}{I \cdot J} \sum_{j=1}^J \sum_{i=1}^I A(i,j), \quad \bar{B} = \frac{1}{I \cdot J} \sum_{j=1}^J \sum_{i=1}^I B(i,j) \\
\sigma_A^2 &= \frac{1}{I \cdot J} \sum_{j=1}^J \sum_{i=1}^I (A(i,j) - \bar{A})^2, \quad \sigma_B^2 = \frac{1}{I \cdot J} \sum_{j=1}^J \sum_{i=1}^I (B(i,j) - \bar{B})^2
\end{aligned} \tag{1}$$

Prostředí Matlab tuto funkci obsahuje již vestavěnou, avšak pro výpočet v jazyce Python byl použit veřejně dostupný skript uživatele [Sabrewarrior](#) z [Github.com](#) (příloha 3), který představuje přímou implementaci funkce `normxcorr2` z Matlabu.

Tato funkce vrací nové dvourozměrné pole, kde každý prvek má nově vypočtenou hodnotu korelačního koeficientu C (obr. 4).



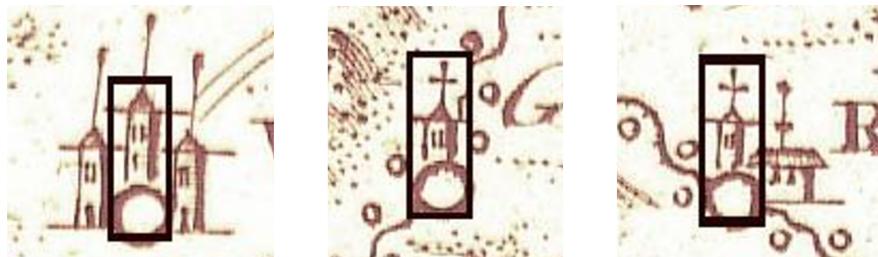
Obrázek 4: výřez vypočteného pole s hodnotami korelačního koeficientu (světlá místa odpovídají vyšším hodnotám C)

Následně se pouze určí pixely, které mají vyšší hodnotu, než je minimální stanovená hodnota korelačního koeficientu.

```
def find_all_symbol_coor(corr_array):      # finds all coordinates with C higher
    coordinate_list = []                   # than the set valid_corr_value
    for r, row in enumerate(corr_array):
        for c, value in enumerate(row):
            if value > valid_corr_value:
                coordinate_list.append([r, c])
    return coordinate_list
```

Zde byla tato hodnota `valid_corr_value` = 0,62 určena empiricky, tak aby byl minimalizován počet chybně identifikovaných objektů a maximalizován počet správně nalezených.

Za úspěšnou identifikaci kostela bylo považováno nalezení znaku jak pro obec s kostelem, tak pro obec s kostelem a zámkem¹, naopak vyhodnocení městečka s trhy bylo brané za nežádoucí. Při nastavené hodnotě $C = 0,62$ k tomuto místy stále dochází, ale při zvýšení prahové hodnoty koeficientu bylo zanedbané příliš velké množství obcí s kostely, takže to bylo tolerováno.



Obrázek 5: porovnání nalezených symbolů – vlevo (městečko s trhy), uprostřed (obec s kostely), vpravo (obec s kostely a zámkem)

¹K tomuto bylo přikročeno vzhledem k nejasnosti zadání, zda je vyhledávaná pouze „obec s kostelem“, či existence kostelů, a vzhledem k tomu, že v legendě Müllerovy mapy je odlišena „obec s kostelem a zámkem“ od pouhé „obce se zámkem“, nebyla existence zámku v daných obcích brána jako závadná. Legenda Müllerovy mapy byla převzata z <https://www.muzeummap.cz/prvni-vystava-v-muzeu-nejkrasnejsi-barokni-mapy-cech-a-moravy-jan-krystof-muller-barokni-kartograf/>

Nejčastěji docházelo k záměně mezi obcí s kostely a obcí s kostely a zámkem, avšak oba dva symboly se v této práci braly jako úspěšné. Každopádně při zvolené hodnotě $C = 0,62$ docházelo k záměně s městečky s trhy (obr. 5) a pří navýšení této hodnoty naopak nebyly registrovány všechny obce s kostely. Proto bylo pár těchto výskytů tolerováno.

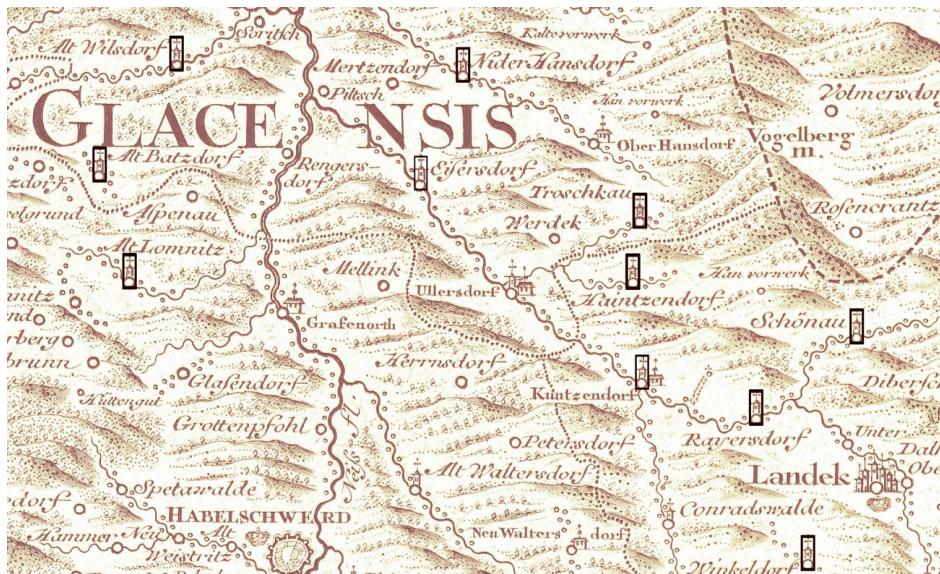
Souřadnice byly prořeđeny o vícenásobné opakování a byly uloženy do souboru formátu `csv` (příloha 1). Toho bylo docíleno tak, že byl nejprve vytvořen nový seznam pro výsledné souřadnice. Následně byly procházeny jednotlivé nalezené objekty a pokud nebyly poblíž žádného z bodů v prořeđeném seznamu souřadnic, byly tam přidány. Vzdálenost pro posouzení, zda se jedná o duplicitu, byla zvolena jako poloviční rozměry vzorové kresby kostela v odpovídajících souřadnicových osách.

```
def filter_duplicates(coordinate_list):
    # filters the coordinate list to remove duplicates that are too close
    # to each other and returns new filtered list
    filtered_list = []

    for pair in coordinate_list:
        is_duplicate = False
        for filt_pair in filtered_list:
            if abs(pair[0]-filt_pair[0]) < sym_height/2 and ...
               abs(pair[1]-filt_pair[1]) < sym_width/2:
                is_duplicate = True
                break
        if not is_duplicate:
            filtered_list.append(pair)

    return filtered_list
```

Na závěr byla tato místa vykreslena do původní mapy (obr. 6).



Obrázek 6: výřez s nalezenými objekty

2.2 Část B

Tato část byla vypočtena především v programu Matlab, avšak k vytvoření výsledné georeferencované vrstvy ve formátu GeoPackage byl použit i programovací jazyk Python.

Výpočetní postup:

1. úprava hodnot původního rastru
2. použití vyhlazovacího filtru (`imgaussfilt()`)
3. segmentace za pomocí metody k-Means
4. morfologické úpravy (vyplnění děr, odebrání drobných lesních ploch)
5. export souřadnic lesní plochy
6. georeferencování výsledného rastru lesních ploch (Python)
7. export do GeoPackage v UTM (Python)

2.2.1 Úprava hodnot původního rastru

Nejprve byla byla úloha řešena klasifikací původního rastru za použití barevného prostoru CIELAB a Standard deviation filtru nebo průměrovacího filtru, leč výsledky byly neuspokojivé. Byla tedy zvolena „prostější“ metoda, a to použití původního barevného prostoru RGB a vytvoření nového prvotního rastru, kterému byly v základu přiděleny nulové hodnoty. Z původního rastru byly určeny hodnoty barev připadající na lesní plochy a vrstevnice. Následně byl rastr procházen pixel po pixelu a jestliže se hodnota pixelu pohybovala v těchto mezích, byla mu přidělena hodnota 255 v rastru novém (viz ukázka kódu). Tímto byly přibližně vybrány plochy, kterých se budou týkat další výpočty.

```
% creates a new raster of zeros
only_green = zeros([size(I, 1:2)]);

for y = 1:size(I,1)                                % fills the raster with values
    for x = 1:size(I,2)
        if (I(y,x,1) > 182 && I(y,x,1) < 227) && ...
            (I(y,x,2) > 155 && I(y,x,2) < 203) && ...
            (I(y,x,3) > 107 && I(y,x,3) < 153)
            only_green(y, x) = 255;
        elseif (I(y,x,1) > 222 && I(y,x,1) < 238) && ...
            (I(y,x,2) > 230 && I(y,x,2) < 250) && ...
            (I(y,x,3) > 160 && I(y,x,3) < 220)
            only_green(y, x) = 255;
        elseif (I(y,x,1) > 240 && I(y,x,1) < 250) && ...
            (I(y,x,2) > 240 && I(y,x,2) < 253) && ...
            (I(y,x,3) > 200 && I(y,x,3) < 220)
            only_green(y, x) = 255;
    end
end
end
```

Pro úsporu času a výsledné georeferencování byl rastr oříznut pouze na velikost mapového okna.

Byly vytvořeny dvě výstupní varianty:

- **nevyplněná:** obsahuje zhruba ponechané lesní průseky, avšak nedokonale odfiltrovává popis a souřadnicovou síť.
- **vyplněná:** výstup neobsahuje žádná hluchá místa po odstranění popisu, avšak ani lesní průseky, celkový výstup působí kompaktněji a přirozeněji.

Výstupem výpočetního skriptu jsou obě varianty, nejprve je vytvořena vyplněná, která je dále použita pro vylepšení nevyplněné.

2.2.2 Použití vyhlazovacího filtru

V této práci byl zvolen vyhlazovací filtr `imgaussfilt(image, sigma)`, který vyhlazuje snímek s 2-D Gaussovým kernelem o směrodatné odchylce σ .

Zvolená směrodatná odchylka σ se liší dle počítané varianty. Při vyplněné variantě je použit `imgaussfilt(image, 5)`, což způsobí, že osamocené vrstevnice budou rozmazenější a v následujících krocích beze stopy zaniknou, lesní plochy se lépe spojí s překryvnými vrstevnicemi, ale dojde i k převážnému vymízení lesních průseků.

Při nevyplněné variantě je použit `imgaussfilt(image, 3)`, což v tomto případě ponechá tvar lesních průseků, ale nedojde k odfiltrování větších mezer způsobených popisem.

2.2.3 Segmentace metodou k-Means

Metoda k-Means clustering je jedním z algoritmů shlukové analýzy, kde k je počet shluků definovaných svými centroidy, přičemž objekty se iterativně zařazují do těchto shluků, čímž se zároveň mění i poloha centroidu.

```
[L,C] = imsegkmeans(rgb_fade, 2, NumAttempts=10);
```

Počet shluků byl nastaven na 2, jelikož je nutné pouze oddělit tmavou plochu od světlé (lesní plocha) a počet iterací byl zvolen 10.

2.2.4 Morfologické úpravy

Byly použity celkem 3 morfologické metody: `strel(image, type, radius)`, `imclose(image, strel())`, `imfill(binary_image, "holes")`, `bwareaopen(binary_image, pixels)`.

Metoda `strel` vytváří morfologické tvary, pro použití ve funkci `imclose`, jež je následně aplikuje na specifikovaný rastr. V tomto případě s parametrem `disk` funkce `strel` vytvoří kruhy o poloměru 10 pixelů a vyplní jimi okolí lesní plochy, čímž zacelí menší útvary. Tato metoda je použita pouze pro vyplněnou variantu.

V případě druhé, vyplněné, varianty byla využita funkce `imfill`, do níž vstupuje binární obraz, ve kterém s parametrem `"holes"` vyplní nesouvislé oblasti.

Naopak v případě první byla použita `bwareaopen`, která v binárním obrazu odstraní shluky, které jsou menší, než zadaná hodnota (50 pixelů).

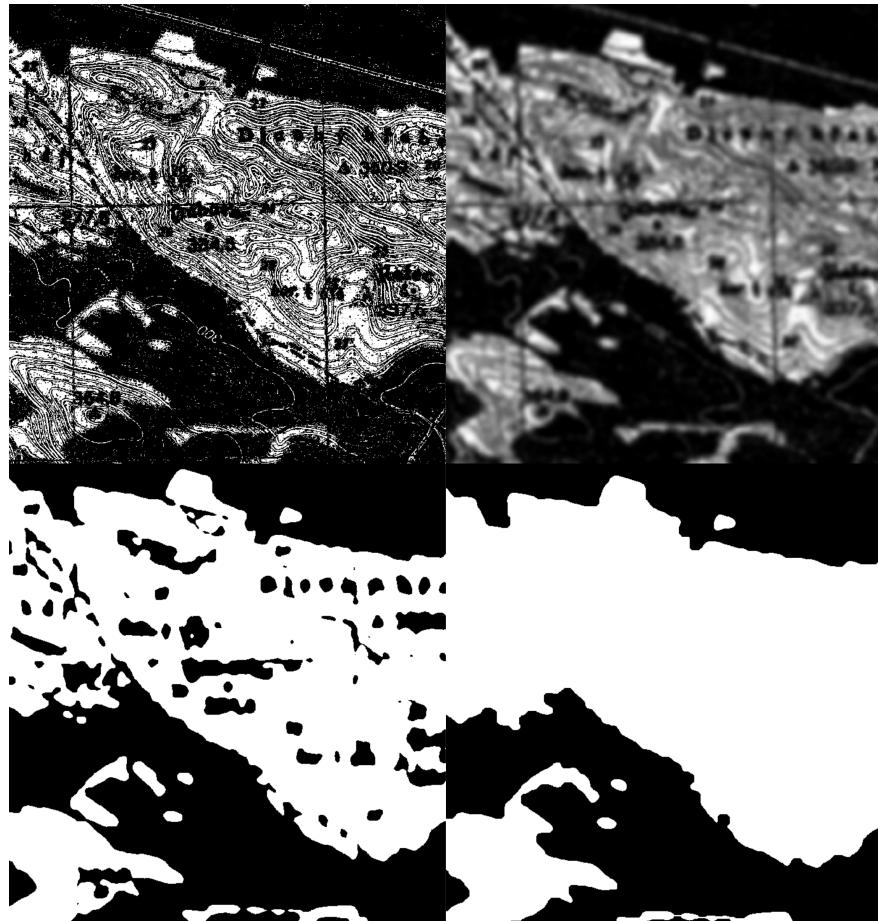
Tato funkce byla také použita k odstranění lesních ploch menších než 50 pixelů u obou variant.

```

%% Filled variant %%
[L,C] = imsegkmeans(rgb_fade, 2, NumAttempts=10); % segmentation to 2 categories
mask = imbinarize(L); % uint8 -> logical values
se = strel('disk', 10); % creates disk object with radiusu = 10
closed = imclose(mask, se); % uses strel object on the mask
filled = imfill(closed, 'holes'); % fills all holes
clean = bwareaopen(filled, 50); % removes small particles under 50 pixels

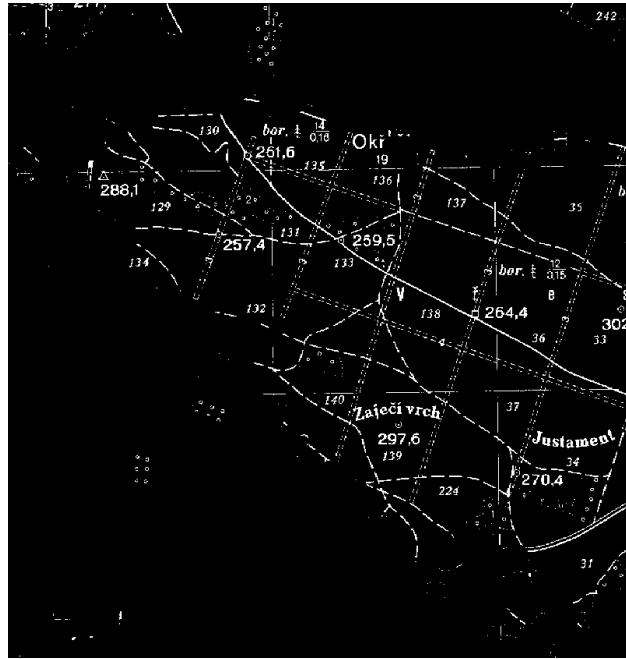
%% Unfilled variant %%
[L,C] = imsegkmeans(rgb_fadef, 2, NumAttempts=10); % segmentation to 2 categories
mask = imbinarize(L); % uint8 -> logical values
se = strel('disk', 2); % creates disk object with radiusu = 2
closedf = imclose(mask, se); % uses strel object on the mask
closedf = ~closedf; % inverts logical values
pfillef = bwareaopen(closedf, 300); % fills holes up to a specified size
pfillef = ~pfillef; % inverts logical values
cleanf = bwareaopen(pfillef, 50); % removes forests smaller than 50 px

```



Obrázek 7: porovnání jednotlivých kroků: vlevo nahoře – vyfiltrování hodnot lesní plochy a vrstevnic; vpravo nahoře – použití `imgaussfilt` funkce; vlevo dole – segmentace pomocí k-Means; vpravo dole – výsledná lesní plocha po morfologických úpravách

Největším nedostatkem první výstupné vrstvy byly reliky mezer v místech, kde se původně nacházel popis nad lesem. Proto byly vybrány pixely s popisem, který kolidoval s lesem, tzn. kontrolou, zda se v pixelu o daných souřadnicích nachází v původním rastru popis a zároveň se ve vyplněné variantě jedná o pixel reprezentující lesní plochu (obr. 8).



Obrázek 8: průnik textu a jiných tmavých pixelů s lesní plochou z vyplněné varianty

```
[r,s] = size(filled); % filter out text
texts_in_forests = zeros(r,s);
for i = 1:r % selects dark pixels that overlap with the filled forest raster
    for j = 1:s
        if (I(i,j,1) >= 0) && (I(i,j,1) <= 130) &&...
            (I(i,j,2) >= 0) && (I(i,j,2) <= 130) &&...
            (I(i,j,3) >= 0) && (I(i,j,3) <= 130) &&...
            (filled(i,j) == 1) % dark pixels in band 1
            % dark pixels in band 2
            % dark pixels in band 3
            % forest in filled raster
        texts_in_forests(i,j) = 1; % assigns true value
    end
end
forests = zeros(r,s); % create a raster with partially filled forests
for i = 1:r
    for j = 1:s % union of two rasters
        if (pfilled(i,j) == 1) || (texts_in_forests(i,j) == 1)
            forests(i,j) = 1; % assigns true value
        end
    end
end
```

V takto nalezených rastrových souřadnicích byla v nevyplněné variantě také vložena hodnota

pro les. Rastr byl následně opět prohnán filtry a morfologickými úpravami, aby se zlepšila jeho kvalita.

2.2.5 Export souřadnic lesní plochy

Proměnná `clean` obsahuje pouze binární hodnoty 0 a 1, kde 1 značí lesní plochu. Přes metodu `find(clean)` byly nalezeny souřadnice všech pixelů, které mají hodnotu 1. Ty byly na závěr uloženy do souboru „lesy.mat“ (příloha 2) metodou `save("lesy.mat", "souřadnice")`. Obdobně postup probíhal i pro nevyplněný rastr s výstupním souborem `lesy_nofill.mat`.

2.2.6 Georeferencování výsledného rastru lesních ploch (Python)

Pro tvorbu GeoPackage byl přesunut zbytek kódu do prostředí jazyka Python, zejména kvůli knihovnám `rasterio`, `geopandas`, `pyproj` a `shapely`, které umožňují snazší georeferenci souborů a práci se souřadnicovými systémy.

Nejprve byla exportována lesní plocha jako snímek z programu Matlab ve formátu TIFF a v programu Python byl tento snímek georeferencován. Ze snímku byly odečteny rohové souřadnice mapového okna,

```
# defines bounding box in geographical coordinates of Krasovsky Ellipsoid (EPSG:4283)
xmin_deg = 14 + 30/60      # 14°30'
ymin_deg = 50 + 35/60      # 50°35'
xmax_deg = 14 + 37.5/60    # 14°37'30"
ymax_deg = 50 + 40/60      # 50°40'
```

které byly spolu s interpolovanými hodnotami celé plochy snímku přetrasformovány affinní transformací ze zeměpisných souřadnic Krasovského elipsoidu (EPSG:4283) do rovinného souřadnicového systému S1942 (EPSG:28403).

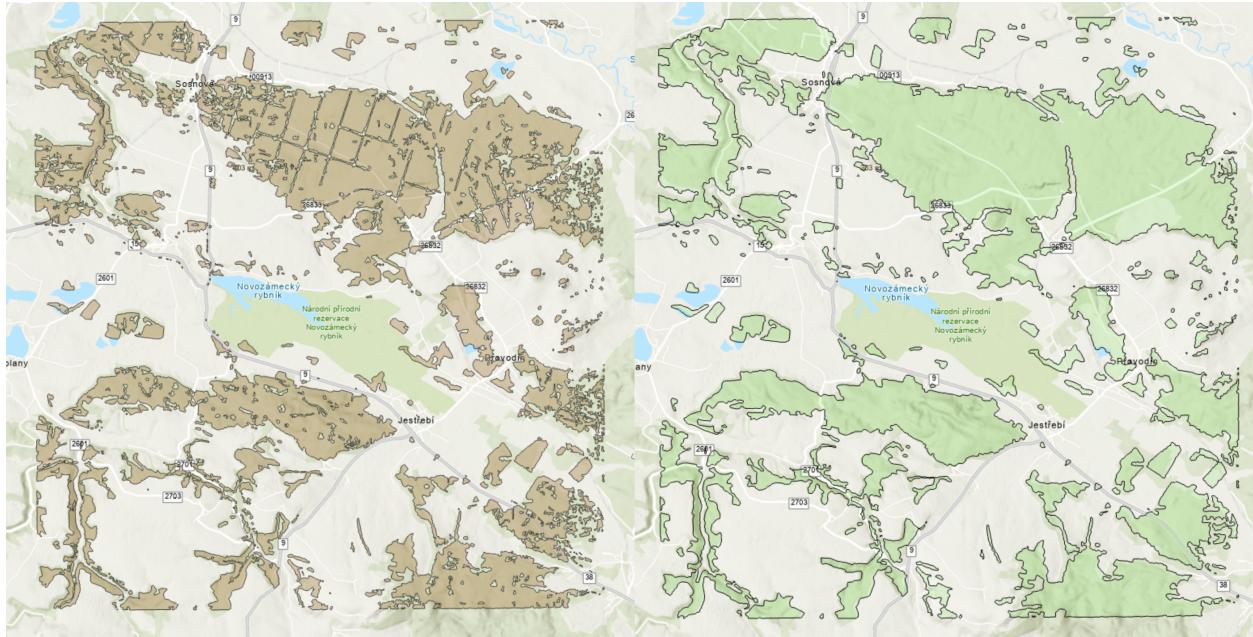
Soubor byl následně uložen ve formátu „GeoTIFF“ pomocí knihovny `rasterio`.

2.2.7 Export do GeoPackage v UTM (Python)

Vestavěnými funkcemi knihovny `rasterio` (metoda `shapes()`) a `shapely` (metoda `shape()`) byly do datových seznamů vloženy jednotlivé shluky lesa, které byly uloženy jako polygony se svou příslušnou rozlohou.

Dále byl vytvořen `GeoDataFrame` knihovnou `geopandas`, do kterého byly tyto geometrie vloženy. Nakonec byla data přetrasformována do UTM (EPSG:32633) a geodatabáze byla exportována ve formátu „GeoPackage“ (ukázka vrstev viz obr. 9).

```
geoms = []                      # creates polygons and saves their area to a new attribute
areas = []
for geom, val in shapes(image, mask=mask, transform=transform):
    geoms.append(shape(geom))
    areas.append(shape(geom).area)           # adds polygon area
gdf = gpd.GeoDataFrame({"area": areas}, geometry=gpd.GeoSeries(geoms, crs=crs))
gdf = gdf.to_crs(utm_epsg)          # transform to UTM
output_gpkg_file = "lesy_polygons_UTM_fill.gpkg"      # saves to GeoPackage
gdf.to_file(output_gpkg_file, layer="lesy_TM25_sk2_fill", driver="GPKG")
```



Obrázek 9: porovnání exportovaných polygonových vrstev lesů v ArcGIS Pro

3 Výstupy

V části A byla na vstupu pouze Müllerova mapa Čech ve formátu JPG a výstupem je seznam souřadnic nalezených kostelů ve formátu CSV, kde každý řádek obsahuje souřadnice x a y . Ve skriptu je možné si odkomentárovat části kódu, které uloží vykreslení rastru v různých částech zpracování (rastr s hodnotami korelačního koeficientu, rastr s nalezenými kostely).

V části B byla na vstupu topografická mapa Československé republiky v měřítku 1 : 25 000 ve formátu JPG. Výstupem skriptu v programu Matlab je seznam souřadnic pixelů nalezené lesní plochy ve formátu .mat (MATLAB Data) a lesní plocha ve formátu TIF (ta je dále nutná pro následující výstup). Výstupem skriptu v jazyce Python je polygonová vrstva lesní plochy ve formátu GeoPackage, ve kterém je každý polygon označen jednoznačným identifikátorem a má uloženou velikost plochy v atributu „area“. Veškeré výstupní soubory jsou ve dvou variantách – se zachovanými průseky, či zcela vyplněné lesní plochy.

Závěr

V části A byl implementován algoritmus pro vyhledání obcí s kostelem v Müllerově mapě Čech. Byly zde provedeny i obě bonusové úlohy, průměrovaný vzor obce s kostelem z 5 různých vzorů a výsledné souřadnice, které jsou proředěny o vícenásobné opakování.

Skript není dokonalý, nenalezne veškeré obce s kostely a vzácně dochází k záměně mezi obcí s kostelem a městečky s trhy, což zapříčinuje podobnost těchto symbolů. Při navýšení hodnoty korelačního koeficientu dochází ke snížené detekci žádaného symbolu, naopak při jejím snížení dochází i k častější detekci nesprávných symbolů. Kompromisem byla použitá hodnota $C = 0,62$.

Část A by se dala vylepšit

- změnou původní podoby rastru, která by mohla zlepšit detekci hledaného symbolu,
- použití jiných rozpoznávacích metod kromě hodnoty korelačního koeficientu
- nebo použitím jiného barvového systému (např. HSL, CIELAB).

V části B byl implementován požadovaný algoritmus obrazové segmentace a byly splněny bonusové úlohy s jistými odchýleními od zadání.

Obraz není segmentovaný s použitím konkrétně Gaborova nebo STD filtru, jelikož oba poskytovaly horší výsledky než nakonec použitý filtr, je zde aplikován `imgaussfilt`, který používá *standard deviations* (směrodatné odchyly) pro vyhlazení rastru.

Druhá bonusová úloha je kompletní, plocha lesa je exportována ve formátu GeoPackage v systému UTM pomocí dalšího skriptu napsaném v prostředí Python.

Vzhledem k velmi neuspokojivým výsledkům získávaným různými postupy byla nakonec úloha řešena z části empiricky a pseudoiterativně, což pravděpodobně nebylo nejfektivnější. Byl použit barevný prostor RGB namísto demonstrovaného CIELAB ze cvičení, a dále se pracovalo pouze z hodnotami, které dosahovali barevných hodnot lesní plochy nebo vrstevnic. Výsledky jsou však srovnatelné a v některých aspektech i lepší.

Pro tuto úlohu byly vytvořeny dvě výstupní varianty – variantu dle zadání, kde jsou ponechány lesní průsek, ale obraz je značně nedokonalý, a vyplněnou variantu, která ztratí informace o lesních průsecích, ale zaplní ostatní díry způsobené překryvnými prvky a výslednou plochu lépe zacelí. Vyplněná varianta více odpovídá skutečné podobě lesní plochy a byla použita pro tvorbu ukázek v této práci.

Část B by se dala vylepšit

- využitím jiného barevného prostoru (např. CIELAB, HSL),
- použití jiných metod obrazové segmentace,
- vytvoření univerzálního skriptu, aby se tato část nemusela spouštět přes program Matlab a následně v prostředí Python,
- změnou původní podoby rastru pro možné zlepšení obrazové segmentace (např. zvýšení kontrastu, ostrosti)
- nebo aplikací vlastnosti lesních průseků, které nejsou uzavřenými mezerami v lesní ploše, ale vlastní geometrii dělí na více částí; bylo by možné tedy vyplnit pouze takové „díry“ v rastru, které by v překrytu nedosáhly okraje celistvé lesní plochy z vyplněného rastru, avšak tato implementace by byla poměrně časově náročná (prohledávání hodnot sousedních pixelů apod.)

Přílohy

- Příloha 1 – seznam souřadnic nalezených obcí s kostely (found_coordinates.csv)
- Příloha 2 – seznam souřadnic lesní plochy (lesy.mat)
- Příloha 3 – skript normxcorr2.py od uživatele Sabrewarrior
- Příloha 4 – skript u2_castA.py pro část A v prostředí Python
- Příloha 5 – skript u2_castB_a.m pro část B v programu Matlab
- Příloha 6 – skript u2_castB_b.py pro část B v prostředí Python

Reference

- [1] Výklad ze cvičení – prof. Ing. Jiří Cajthaml, Ph.D.; Ing. Tomáš Janata, Ph.D.
- [2] 155YGEI Geoinformatika. Online. 2025. Dostupné z: https://geo.fsv.cvut.cz/gwiki/155YGEI_Geoinformatika. [cit. 2025-10-13].

Podepsáno dne 18. 11. 2025 v Praze

Králič Adam, Matějková Barbora