

Vysoká škola

Fakulta aplikovaných věd

Semestrální práce

Automatické prahování obrazu

Metody Otsu a Sauvola

Autor: Jan Čácha
Studijní program: Informatika
Předmět: Zpracování vizuální informace
Akademický rok: 2024/2025

27. března 2025

Obsah

1	Úvod	1
2	Teoretický rozbor	1
2.1	Otsuova metoda	1
2.2	Recursive Otsu metoda	1
2.3	Sauvolova metoda	2
3	Implementace	2
3.1	Použité technologie	2
3.2	Klíčové algoritmy	2
3.2.1	Otsuovo prahování	2
3.2.2	Sauvolovo prahování	3
3.2.3	Rekurzivní Otsu	3
3.2.4	Selektivní bilaterální filtrace	3
4	Výsledky	3
5	Uživatelská příručka	4
5.1	Instalace	4
5.2	Spuštění aplikace	5
5.3	Popis rozhraní	5
5.3.1	Horní panel	5
5.3.2	Střední část	5
5.3.3	Nastavení prahování	5
5.4	Postup práce	5
5.5	Řešení problémů	6
5.6	Doporučené parametry	6
6	Závěr	6

Abstrakt

Tato práce se zabývá implementací dvou metod automatického prahování obrazu - Otsuovy metody a Sauvolova adaptivního prahování. Vytvořili jsme aplikaci v Pythonu s grafickým rozhraním, která umožňuje experimentální porovnání těchto metod. Práce obsahuje teoretický rozbor, popis implementace a diskuzi výsledků.

1 Úvod

Prahování je základní operace v digitálním zpracování obrazu používaná pro separaci objektů od pozadí. Cílem této práce je:

- Implementovat Otsuovu metodu (globální prahování)
- Implementovat Sauvolovu metodu (lokální adaptivní prahování)
- Vytvořit uživatelsky přívětivé GUI pro experimentování
- Vyhodnotit výhody a nevýhody obou metod

2 Teoretický rozbor

2.1 Otsuova metoda

Metoda navržená Nobuyukim Otsuem v roce 1979 hledá optimální prah maximalizací mezitřídního rozptylu.

Matematický základ:

$$\sigma_b^2(t) = w_0(t) \cdot w_1(t) \cdot [\mu_0(t) - \mu_1(t)]^2 \quad (1)$$

Kde:

- w_0, w_1 jsou pravděpodobnosti tříd
- μ_0, μ_1 jsou střední hodnoty tříd
- t je testovaný práh

2.2 Recursive Otsu metoda

Rozšíření klasické Otsuovy metody [?] řeší problémy s:

- Slabými tahy písma (faint strokes)
- Prosakováním z druhé strany (bleed-through)
- Nerovnoměrným osvětlením (non-uniform illumination)

Algoritmus pracuje v těchto krocích:

1. Odhad pozadí pomocí medianového filtru (velikost okna 21×21)
2. Odstranění pozadí subtrakcí

3. Bilaterální filtrace pro redukci šumu ($\sigma_s = 10$, $\sigma_r = 2$)

4. Rekurzivní prahování:

$$T_k = \text{Otsu}(T_{k-1}, 255) \quad \text{s kritérii zastavení } d_1 = 2, d_2 = 26 \quad (2)$$

5. Selektivní bilaterální filtrace (odlišné parametry pro text a pozadí)

6. Prahování s hysterezí

Klíčovou inovací je rekurzivní aplikace Otsuovy metody na postupně se zužující intenzitní rozsah, což umožňuje zač

2.3 Sauvolaova metoda

Adaptivní metoda vyvinutá Sauvolou a Pietikäinenem počítá lokální práh podle vzorce:

$$T(x, y) = \mu(x, y) \left[1 + k \left(\frac{\sigma(x, y)}{R} - 1 \right) \right] \quad (3)$$

Parametry:

- k - citlivost na kontrast (0.1-0.5)
- R - normalizační konstanta (typicky 128)

3 Implementace

3.1 Použité technologie

- Python 3 s knihovnami: OpenCV, NumPy, PIL
- Tkinter pro grafické rozhraní
- Matplotlib pro zobrazování histogramů

3.2 Klíčové algoritmy

3.2.1 Otsuovo prahování

```
def otsu_threshold(image):  
    hist = np.histogram(image, bins=256)[0]  
    prob = hist / hist.sum()  
    max_var, optimal = 0, 0  
    for t in range(1, 256):  
        w0, w1 = prob[:t].sum(), prob[t:].sum()  
        mu0 = (np.arange(t)*prob[:t]).sum()/w0  
        mu1 = (np.arange(t, 256)*prob[t:]).sum()/w1  
        var = w0*w1*(mu0-mu1)**2  
        if var > max_var:  
            max_var, optimal = var, t  
    return optimal
```

3.2.2 Sauvolovo prahování

```
def sauvola(image, window=15, k=0.2, R=128):
    pad = window//2
    padded = np.pad(image, pad, mode='reflect')
    result = np.zeros_like(image)
    for i in range(image.shape[0]):
        for j in range(image.shape[1]):
            window = padded[i:i+window, j:j+window]
            mean, std = window.mean(), window.std()
            T = mean*(1 + k*(std/R - 1))
            result[i,j] = 255 if image[i,j] > T else 0
    return result
```

3.2.3 Rekurzivní Otsu

```
def recursive_otstu(image, levels=3, d1=2, d2=26):
    thresholds = []
    prev_thresh = 0
    for _ in range(levels):
        current_image = image if not thresholds else image[image >= prev_thresh]
        thresh = otsu_threshold(current_image)
        if thresholds and (thresh-prev_thresh < d1 or thresh-prev_thresh > d2):
            break
        thresholds.append(thresh)
        prev_thresh = thresh
    return thresholds
```

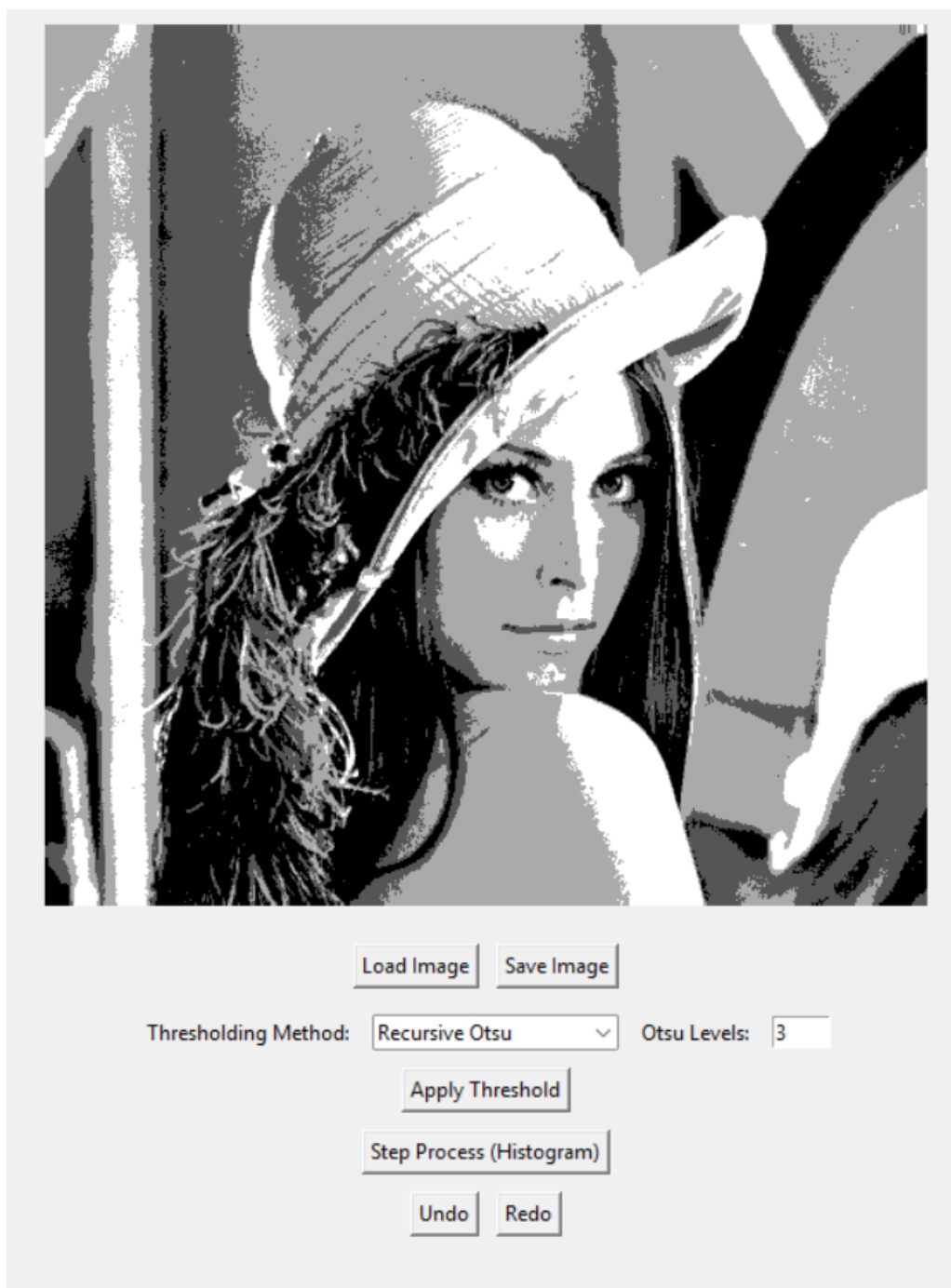
3.2.4 Selektivní bilaterální filtrace

```
def selective_bilateral_filter(image, mask):
    fg = cv2.bilateralFilter(image, -1, 2, 2) # Konzervativní pro text
    bg = cv2.bilateralFilter(image, -1, 10, 3) # Agresivnější pro pozadí
    return np.where(mask > 0, fg, bg)
```

4 Výsledky

Tabulka 1: Porovnání metod

Metoda	Výhody	Nevýhody
Otsu	Rychlá, dobrá pro bimodální histogram	Špatně pracuje s nerovnoměrným osvětlením
Recursive Otsu	Zachycuje slabé tahy, robustní k degradaci	Náročnější na výpočet
Sauvola	Adaptivní k místním podmínkám	Pomalá výpočetně náročná



Obrázek 1: Příklad prahování Recursive Otsu

5 Uživatelská příručka

5.1 Instalace

1. Instalace Pythonu 3.8 nebo novějšího z <https://www.python.org/downloads/>
2. Instalace potřebných knihoven:

```
pip install opencv-python numpy pillow matplotlib scikit-image
```

3. Stažení zdrojového kódu aplikace z repozitáře:

5.2 Spuštění aplikace

ZVI-Thresholding-RecursiveOtsu-Sauvola
python threshold.py

5.3 Popis rozhraní

Aplikace obsahuje následující ovládací prvky:

5.3.1 Horní panel

- **Načíst obrázek** - Tlačítko pro výběr vstupního obrázku (formáty: PNG, JPG, BMP)
- **Uložit obrázek** - Uložení výsledku prahování

5.3.2 Střední část

- **Zobrazovací panel** - Zobrazuje původní a upravený obrázek
- **Undo/Redo** - Vrácení nebo obnovení předchozí operace

5.3.3 Nastavení prahování

- **Metoda:**
 - Otsu - Globální metoda s volbou počtu úrovní (2-4)
 - Sauvola - Lokální metoda s nastavením:
 - * Velikost okna (liché číslo ≥ 3)
 - * Parametr k (0.1-0.5)
 - * Parametr R (doporučeno 128)
- **Histogram** - Zobrazení histogramu aktuálního obrázku

5.4 Postup práce

1. Načtěte vstupní obrázek pomocí tlačítka *Načíst obrázek*
2. Zvolte metodu prahování:
 - Pro Otsu: Nastavte počet úrovní (pro binární prahování zvolte 2)
 - Pro Sauvola: Experimentujte s velikostí okna a parametry k, R
3. Klikněte na *Aplikovat prahování*
4. Prohlédněte si výsledek a v případě potřeby upravte parametry
5. Pro zobrazení histogramu klikněte na *Histogram*
6. Výsledek uložte pomocí *Uložit obrázek*

5.5 Řešení problémů

- **Chybějící knihovny:** Spusťte příkaz pro instalaci knihoven
- **Pomalé zpracování:** Pro Sauvolovu metodu zvolte menší okno
- **Špatné výsledky:**
 - U Otsu: Zkontrolujte, zda má obrázek bimodální histogram
 - U Sauvola: Upravte parametr k (zvýšit pro nízký kontrast)
- **Nefungují tlačítka Undo/Redo:** Operace se ukládají až po dokončení

5.6 Doporučené parametry

Tabulka 2: Doporučené hodnoty parametrů

Typ obrazu	Metoda	Parametry
Dobře osvětlený dokument	Otsu	Úrovně: 2
Dokument s nerovnoměrným osvětlením	Sauvola	Okno: 15-25, $k=0.3$, $R=128$
Obrázek s šumem	Sauvola	Okno: 15, $k=0.2$, $R=64$

6 Závěr

Práce úspěšně implementovala obě metody prahování. Z výsledků vyplývá:

- Otsuova metoda je vhodná pro obrazy s rovnoměrným osvětlením
- Sauvolova metoda lépe zpracovává reálné dokumenty
- GUI aplikace umožňuje snadné experimentování

Možná vylepšení:

- Optimalizace Sauvolovy metody pomocí integrálních obrazů
- Přidání dalších metod prahování

Literatura

1. Otsu, N. (1979). A Threshold Selection Method from Gray-Level Histograms
2. Sauvola, J. (2000). Adaptive Document Image Binarization
3. Gonzalez, R. C. (2008). Digital Image Processing
4. Nina, O., Morse, B., Barrett, W. (2010). *A Recursive Otsu Thresholding Method for Scanned Document Binarization*. Brigham Young University.