VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

ÚSTAV MATEMATIKY
INSTITUTE OF MATHEMATICS

SLEDOVÁNÍ OBJEKTŮ V OBRAZECH POŘÍZENÝCH VYSOKORYCHLOSTNÍ KAMEROU
OBJECT TRACKING IN HIGH-SPEED CAMERA IMAGES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Michal Myška**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Pavel Štarha, Ph.D.**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Ústav matematiky
Student:	**Bc. Michal Myška**
Studijní program:	Aplikované vědy v inženýrství
Studijní obor:	Matematické inženýrství
Vedoucí práce:	**doc. Ing. Pavel Štarha, Ph.D.**
Akademický rok:	2019/20

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Sledování objektů v obrazech pořízených vysokorychlostní kamerou

Stručná charakteristika problematiky úkolu:

Vysokorychlostní kamery umožňují zaznamenat pozice rychle se pohybujících objektů v jednotlivých časových intervalech. Tímto lze rekonstruovat jejich trajektorii a orientaci v prostoru. V našem případě se bude jednat o mikroskopická skleněná vlákna pohybujících se v modelu lidských plic.

Cíle diplomové práce:

Seznámit se základními numerickými metodami zpracování obrazové informace.
Provést segmentaci dat a identifikaci sledovaného objektu.
Nalézt trasu a orientaci pohybujícího se objektu.
Naprogramovat jednoúčelovou aplikaci pro dané zpracování dat.

Seznam doporučené literatury:

MARTIŠEK, Dalibor. Matematické principy grafických systémů. Brno: Littera, 2002, 278 s. ISBN 80-857-6319-2.

KLÍMA, Miloš. Zpracování obrazové informace. V Praze: České vysoké učení technické, 1996. ISBN 8001014363.

PRATT, William K. Digital Image Processing (Third Edition) PIKS Inside [online]. 3rd ed. New York: Wiley-Interscience, 2001 [cit. 2014-08-07]. ISBN 04-712-2132-5. Dostupné z: http://www.csupomona.edu/~kding/materials/Digital Image Processing - Third Edition - William K.Pratt.pdf.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2019/20

V Brně, dne

L. S.

prof. RNDr. Josef Šlapal, CSc. ředitel ústavu	doc. Ing. Jaroslav Katolický, Ph.D. děkan fakulty

Abstrakt

Tato diplomová práce se zabývá sledováním objektů v obrazech pořízených vysokorychlostní kamerou, u kterých hledáme jejich trajektorii a orientaci. Práce obsahuje základní matematickou teorii spojenou s tímto problémem a jsou taktéž uvedeny metody používané pro zpracování obrazu. Hlavním výsledkem práce je jednoúčelová aplikace, která má uživatelské prostředí a pomocí které můžeme měřit sledované parametry jednotlivých objektů.

Abstract

This master thesis is dealing with object tracking in high-speed camera images, within what we are trying to find their trajectory and orientation. The mathematical theory associated with this problem as well as the methods used fo image processing are described here. The main outcome is an application with a user interface through which we can calculate the desired parameters of the individual objects.

klíčová slova

sledování objektů, momentová metoda, konvoluce, zpracování obrazu

keywords

object tracking, moment method, convolution, image processing

Myška, M.: *Sledování objektů v obrazech pořízených vysokorychlostní kamerou.*, Brno, 2020. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/121639. Diplomová práce, Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav matematiky, 2020. 77s. Vedoucí diplomové práce doc. Ing. Pavel Štarha, Ph.D.

Prohlašuji, že jsem diplomovou práci *Sledování objektů v obrazech pořízených vysokorychlostní kamerou* vypracoval samostatně pod vedením doc. Ing. Pavla Štarhy, Ph.D. s použitím materiálů uvedených v seznamu literatury.

V Brně dne ....

Michal Myška

Rád bych tímto poděkoval vedoucímu své diplomové práce panu doc. Ing. Pavlu Štarhovi, Ph.D. za trpělivost a především cenné rady, díky kterým jsem byl schopen tuto diplomovou práci dokončit. Zároveň bych rád poděkoval své rodině za podporu během celého studia.

Michal Myška

Obsah

Úvod

Tato práce se zabývá sledováním mikroskopických skleněných vláken, která byla vytvořena ze skelné vaty Supafil® Loft a jejichž diametr a délka byly postupně $3.8 \pm 1.4 \mu m$ a $34.1 \pm 19.0 \mu m$. Jejich pohyb v replice lidské průdušnice byl zaznamenáván pomocí vysokorychlostní kamery Photron SA-Z. Vdechování vláken může potencionálně vést k zánětu plic a intersticii, což při dlouhodobějším vystavení může vést až k rakovině plic, malignímu mezoteliomu nebo plicní a pleurální fibróze. Toxicita vláken silně souvisí s jejich dimenzemi víc než s jejich chemickým složením. Abychom redukovali riziko při práci s těmito vlákny, je nutné znát, co se s jednotlivými vlákny při vdechnutí stane v závislosti na jejich fyzikálních charakteristikách a podmínkách vdechnutí. Pro tyto účely byl sestaven speciální mechanismus, který simuluje a zaznamenává průtok vláken[4].

Hlavním cílem této práce pak byla identifikace těchto vláken v obrazech dané kamery a určení trajektorie a orientace během pohybu, a to na základě praktického využití numerických metod zpracování obrazové informace. K tomuto účelu byla vytvořena jednoúčelová aplikace v programovacím jazyce Python 3.7.

Studium těchto vláken má toxikologický charakter, tedy snažíme se zjistit dopad přítomnosti těchto vláken v našich dýchacích cestách, z tohoto důvodu je důležité zabývat se jejich pohybem a analyzovat je. Do budoucna je u vláken podobného typu potenciál farmaceutický, kdy tato vlákna mohou být nosiči určitých farmaceutických látek a za předpokladu znalosti jejich pohybu mohou být tyto látky dopravovány na specifická místa.

Samotná práce je rozdělena do čtyř kapitol. V první kapitole popíšeme obecné numerické metody a dále pak odvodíme jednotlivé vztahy pro využití při zpracovávání obrazové informace.

V druhé části se budeme nejprve věnovat zavedení obrazové matice a poté si popíšeme běžně používané metody pro filtraci obrazu, především pak pro filtraci šumu. Dále se zaměříme na segmentaci obrazu a operace spojené s binárním obrazem. V neposlední řadě si zavedeme také tzv. sekvenční filtraci, díky které budeme schopni lépe rozeznat jednotlivá vlákna v obrazech.

Předposlední část se týká hlavního cíle této práce, a to vytvořené jednoúčelové aplikaci. Kapitola poskytuje popis postupu zpracování vstupních dat, spolu s porovnáním výstupů aplikace, ve které jsme využili několika metod, popsaných dříve v textu.

Na závěr se zaměříme na shrnutí výsledků, které jsme získali námi vytvořenou aplikací, a to především pak na jejich vizualizaci.

1 Matematický aparát

V této kapitole se zaměříme na obecné numerické metody a odvodíme si vztahy pro jejich použití ve zpracování obrazu. Zdrojem informací pro tuto kapitolu byly [1][3][6][16][22] [26][27].

1.1 Interpolace kubickým splajnem

Mějme zadaných $n+1$ uzlových bodů $x_0 < x_2 < \cdots < x_n$ a funkční hodnoty v těchto bodech $f_0, f_1, \ldots, f_n$. Splajnem rozumíme funkci $S(x)$ takovou, pro kterou platí $S(x_i) = f_i$, kde $i = 0, \ldots, n$, kdy tato funkce je po částech polynom a splňuje podmínky hladkosti. Nejběžněji se používá interpolace kubickým polynomem, tedy kubickým interpolačním splajnem.

Splajn k-tého řádu má obecně spojité derivace do $k-1$ řádu, tedy v případě kubického splajnu má interpolační křivka spojité první i druhé derivace. Odtud nám plynou základní podmínky kubického splajnu

1. $S_i = f_i(x_i), \quad \text{pro } i = 0, \ldots, n$
2. $S_i(x_{i+1}) = S_{i+1}(x_{i+1}) \quad \text{pro } i = 0, \ldots, n-2$
3. $S_i'(x_{i+1}) = S_{i+1}'(x_{i+1}) \quad \text{pro } i = 0, \ldots, n-2$
4. $S_i''(x_{i+1}) = S_{i+1}''(x_{i+1}) \quad \text{pro } i = 0, \ldots, n-2.$

Dále zavádíme okrajové podmínky, které záleží na typu kubického splajnu. V našem případě se omezíme na tzv. *přirozený kubický splajn* pro který platí

$$S''(x_0) = S''(x_n) = 0.$$

Kubický splajn $S(x)$ hledáme na jednotlivých intervalech $\langle x_i, x_{i+1}\rangle$, kde $i = 0, \ldots, n$ obecně ve tvaru

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3.$$

Z podmínek kubického splajnu a okrajové podmínky můžeme odvodit soustavu rovnic s neznámými parametry c_i, kde $i = 0, \ldots, n$ ve tvaru

$$h_{i-1}c_{i-1} + 2(h_{i-1} + h_i)c_i + h_i c_{i+1} = 3\left(q_i - q_{i-1}\right), i = 1, \ldots, n-1, c_0 = c_n = 0$$

kde

$$\begin{aligned} h_i &= x_{i+1} - x_i, \\ q_i &= \frac{f(x_{i+1}) - f(x_i)}{h_i}, \quad i = 0, \ldots, n-1. \end{aligned}$$

Výsledný tvar této soustavy po rozepsání a dosazení c_1 a c_n má podobu

$$\begin{bmatrix} 2(h_0+h_1) & h_1 & & & \\ h_1 & (h_1+h_2) & h_2 & & \\ & & \ddots & & \\ & & h_{n-2} & 2(h_{n-2}+h_{n-1}) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} 3\left(q_1 - q_0\right) \\ 3\left(q_2 - q_1\right) \\ \vdots \\ 3\left(q_{n-1} - q_{n-2}\right) \end{bmatrix}$$

Jedná se tedy o soustavu s třídiagonální maticí, kterou lze vyřešit například pomocí Gaussovy eliminační metody, která je popsána například v [8].

Koeficienty b_i a d_i pak dopočítáme pomocí vztahů

$$b_i = \frac{f(x_{i+1}) - f(x_i)}{h_i} - \frac{c_{i+1} + 2c_i}{3} h_i$$
$$d_i = \frac{c_{i+1} - c_i}{3h_i}, \quad i = 0, \ldots, n-1.$$

Pro naše účely budeme interpolovat body v rovině, které by měly náležet rovinné křivce. Budeme tedy hledat křivku danou parametricky, která bude procházet body o souřadnicích $[x_i, y_i]$. Pro tyto účely použijeme interpolaci kubickým splajnem po složkách. Tedy

$$S_i^x(\varphi) = a_i^x + b_i^x(\varphi - x_i) + c_i^x(\varphi - x_i)^2 + d_i^x(\varphi - x_i)^3,$$
$$S_i^y(\varphi) = a_i^y + b_i^y(\varphi - y_i) + c_i^y(\varphi - y_i)^2 + d_i^y(\varphi - y_i)^3,$$

kde $[\varphi, x_i]$, resp. $[\varphi, y_i]$ jsou uzlové body, pro které platí $x_i = x(\varphi)$, resp. $y_i = y(\varphi)$ a $\varphi \in \langle \varphi_{i-1}, \varphi_i \rangle$, $i = 1, \ldots, n$. Koeficienty b_i^x, c_i^x, d_i^x a b_i^y, c_i^y, d_i^y určíme obdobným způsobem a získáme

$$x(\varphi) = \bigcup_{i=0}^{n} S_i^x(\varphi)$$
$$y(\varphi) = \bigcup_{i=0}^{n} S_i^y(\varphi),$$

kde $\varphi \in \langle 0, \varphi_n \rangle$. Interpolační křivka, která prochází rovinnými body je pak dána jako

$$\theta(\varphi) = [x(\varphi), y(\varphi)]\,.$$

1.2 Metoda nejmenších čtverců

V mnoha situacích se dostaneme do situace, kdy pro nás není žádoucí, aby interpolační křivka procházela naměřenými referenčními body, jelikož můžou být jistým způsobem ovlivněny. V takových případech se budeme bavit o aproximaci, jejímž typickým příkladem je metoda nejmenších čtverců, která je metodou prokládání dat.

Pro naše účely budeme aproximovat body těžiště pohybujícího se objektu, proto se zaměříme na případ, kdy máme křivku zadanou parametricky.

Uvažujme n bodů o souřadnicích $[x_i(t_i), y_i(t_i)]$, $t_i \in \mathbb{R}$, kde $i = 1, \ldots, n$ a tvar složené lineární aproximační křivky λ

$$\lambda(\gamma, \delta, t) = [\gamma_1 f_1(t) + \gamma_2 f_2(t) + \cdots + \gamma_k f_k(t), \delta_1 g_1(t) + \delta_2 g_2(t) + \cdots + \delta_k g_k(t)],$$

kde k je počet jednotlivých funkcí $f_i(t)$ a $g_i(t)$ a $t \in \mathbb{R}$. Jednotlivé funkce $f_i(t)$ jsou lineárně nezávislé, stejně jako $g_i(t)$. Proměnné $\gamma = (\gamma_1, \ldots, \gamma_k)$ a $\delta = (\delta_1, \ldots, \delta_k)$ hledáme za účelem minimalizace kvadrátu vzdáleností $\nu(\gamma, \delta)$ bodů $[x_i(t), y_i(t)]$ od aproximační křivky. Tedy chceme najít

$$\min_{\gamma,\delta} \nu(\gamma, \delta) = \min_{\gamma,\delta} \sum_{i=1}^{n} \varrho^2 \left(\lambda(\gamma, \delta, t_i), [x_i(t_i), y_i(t_i)]\right),$$

kde ϱ je euklidovská metrika. Aproximační křivku upravíme

$$\lambda(\gamma, \delta, t) = [\varphi_x(\gamma, t), \varphi_y(\delta, t)],$$

tedy platí

$$\begin{aligned} \varphi_x(\gamma, t) &= \gamma_1 f_1(t) + \gamma_2 f_2(t) + \cdots + \gamma_k f_k(t), \\ \varphi_y(\delta, t) &= \delta_1 g_1(t) + \delta_2 g_2(t) + \cdots + \delta_k g_k(t). \end{aligned} \tag{1.1}$$

Nyní můžeme postupně upravovat

$$\begin{aligned} \min_{\gamma,\delta} \nu(\gamma, \delta) &= \min_{\gamma,\delta} \sum_{i=1}^{n} ((\varphi_x(\gamma, t) - x(t_i))^2 + (\varphi_y(\delta, t) - y(t_i))^2) \\ \min_{\gamma,\delta} \nu(\gamma, \delta) &= \min_{\gamma,\delta} \left(\sum_{i=1}^{n} (\varphi_x(\gamma, t) - x(t_i))^2 + \sum_{i=1}^{n} (\varphi_y(\delta, t) - y(t_i))^2 \right) \\ \min_{\gamma,\delta} \nu(\gamma, \delta) &= \min_{\gamma} \sum_{i=1}^{n} (\varphi_x(\gamma, t) - x(t_i))^2 + \min_{\delta} \sum_{i=1}^{n} (\varphi_y(\delta, t) - y(t_i))^2. \end{aligned}$$

Jak můžeme tedy vidět, pro výpočet aproximační křivky lze využít výpočtu pro každou složku zvlášť a tedy řešíme výpočet aproximační křivky jedné proměnné, který si teď přiblížíme.

Uvažujme aproximační funkci ve tvaru

$$\varphi(\gamma, x) = \gamma_1 f_1(x) + \gamma_2 f_2(x) + \cdots + \gamma_k f_k(x),$$

kde k je počet funkcí $f_i(x)$, které jsou lineárně nezávislé a mějme opět n bodů se souřadnicemi $[x_i, y_i]$, $i = 1, \ldots, n$. Proměnné $\gamma = (\gamma_1, \ldots, \gamma_k)$ se snažíme určit tak, aby kvadrát odchylek $\upsilon(\gamma)$ jednotlivých bodů od aproximační křivky byl minimální. To znamená

$$\min_{\gamma} \upsilon(\gamma) = \min_{\gamma} \sum_{i=1}^{n} (\varphi(\gamma, x_i) - y(y_i))^2. \tag{1.2}$$

Řešení minimalizační úlohy (1.2) musí splňovat podmínku pro extrém

$$\frac{\partial \upsilon}{\partial \gamma_k} = 0, \quad k \ = 1, \ldots, n.$$

Pokud provedeme naznačené derivování a dostaneme

$$\begin{aligned} \frac{\partial \upsilon}{\partial \gamma_1} &= \sum_{i=1}^{n} 2(\varphi(\gamma, x_i) - y(y_i)) f_1(x_i) = 0 \\ \frac{\partial \upsilon}{\partial \gamma_2} &= \sum_{i=1}^{n} 2(\varphi(\gamma, x_i) - y(y_i)) f_2(x_i) = 0 \\ &\vdots \\ \frac{\partial \upsilon}{\partial \gamma_k} &= \sum_{i=1}^{n} 2(\varphi(\gamma, x_i) - y(y_i)) f_k(x_i) = 0. \end{aligned}$$

Odkud po úpravě dostaneme k lineárních rovnic o k neznámých, které můžeme rozepsat do matice jako

$$\begin{bmatrix} \sum_{i=1}^{n} f_{i,1}^2 & \sum_{i=1}^{n} f_{i,2}f_{i,1} & \dots & \sum_{i=1}^{n} f_{i,k}f_{i,1} \\ \sum_{i=1}^{n} f_{i,1}f_{i,2} & \sum_{i=1}^{n} f_{i,2}^2 & \dots & \sum_{i=1}^{n} f_{i,k}f_{i,2} \\ \vdots & & \ddots & \vdots \\ \sum_{i=1}^{n} f_{i,1}f_{i,k} & \sum_{i=1}^{n} f_{i,2}f_{i,k} & \dots & \sum_{i=1}^{n} f_{i,k}^2 \end{bmatrix} \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_k \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} y_i f_{i,1} \\ \sum_{i=1}^{n} y_i f_{i,2} \\ \vdots \\ \sum_{i=1}^{n} y_i f_{i,k} \end{bmatrix},$$

kde $f_j(x_i) = f_{i,j}, j = 1, \dots, k$.

1.3 Integrální transformace

Obecně můžeme integrální transformaci funkce dvou proměnných $f(x, y)$ zapsat jako

$$F(\xi, \eta) = c \iint_{\mathbb{R}^2} f(x, y)\psi_{\xi,\eta}(x, y)\mathrm{d}x\mathrm{d}y$$

a zpětnou integrální transformaci

$$f(x, y) = c \iint_{\mathbb{R}^2} F(\xi, \eta)\psi^{-1}_{x,y}(\xi, \eta)\mathrm{d}\xi\mathrm{d}\eta.$$

Funkce $\psi_{\xi,\eta}(x, y)$, resp. $\psi^{-1}_{x,y}(\xi, \eta)$ označují jádro transformace, resp. inverzní jádro transformace.

1.4 Fourierova transformace

Transformací funkce docílíme jejího vyjádření ve výhodnějším tvaru pro určitý typ zpracování. Nejčastěji se setkáváme s Fourierovou transformací, která je základním matematickým nástrojem při zpracování obrazu a jeho analýze. Zaměříme se pouze na případ transformace funkce dvou proměnných, jelikož tou je i obraz.

1.4.1 Spojitá Fourierova transformace

Definice 1.1 (**Fourierova transformace**)**.** Nechť je dána komplexní funkce $f(x, y)$ definovaná na $\mathbb{R}^2$. Fourierovým obrazem funkce $f(x, y)$ nazveme komplexní funkci $F(\xi, \eta)$ reálných proměnných ξ, η , kterou definujeme pomocí integrálu

$$F(\xi, \eta) = \iint_{\mathbb{R}^2} f(x, y)e^{-2\pi i(x\xi+y\eta)}\mathrm{d}x\mathrm{d}y$$

a značíme $\mathcal{F}\{f(x, y)\}$.

Platí tedy, že jádro transformace $\psi_{\xi,\eta}(x, y) = e^{-2\pi i(x\xi+y\eta)}$. Dále si definujme také inverzní (zpětnou) Fourierovu transformaci.

Definice 1.2 (**Inverzní Fourierova transformace**)**.** Nechť je dána komplexní funkce $f(x, y)$ definovaná na $\mathbb{R}^2$ a komplexní funkci $F(\xi, \eta)$, která je jejím Fourierovým obrazem. Pak pro inverzní (zpětnou) Fourierovu transformaci funkce $F(\xi, \eta)$ platí

$$f(x, y) = \iint_{\mathbb{R}^2} F(\xi, \eta) e^{2\pi i(x\xi + y\eta)} \mathrm{d}x\mathrm{d}y$$

a značíme $\mathcal{F}^{-1}\{F(\xi, \eta)\}$.

V tomto případě je inverzní jádro tranformace $\psi^{-1}_{x,y}(\xi, \eta) = e^{2\pi i(x\xi+y\eta)}$. Více o spojité Fourierově transformaci je možné nalézt například v [1] nebo [3].

1.4.2 Diskrétní Fourierova transformace

Při zpracování obrazové informace na počítači dochází k diskretizaci obrazu v prostorových souřadnicích. Obraz je reprezentován maticí (více informací v kapitole (2.1)), u které uvažujme velikost $N \times N$. Z tohoto důvodu zavádíme diskrétní Fourierovu transformaci.

Definice 1.3 (**Diskrétní Fourierova transformace**)**.** Nechť $\mathbf{f}(k, l) : \{0, 1, \ldots, N-1\} \times \{0, 1, \ldots, N-1\} = \{0, 1, \ldots, N-1\}^2 \to \mathbb{C}, N \in \mathbb{N}$. Diskrétní Fourierovým obrazem $\mathbf{f}(k, l)$ nazveme funkci $F(\xi, \eta) : \{0, 1, \ldots, N-1\}^2 \to \mathbb{C}$, která je dána jako

$$F(\xi, \eta) = \frac{1}{N} \sum_{\xi=0}^{N-1} \sum_{\eta=0}^{N-1} f(k, l) e^{-i\frac{2\pi}{N}(k\xi + l\eta)}$$

a značíme $\mathcal{D}\{\mathbf{f}(k, l)\}$.

Stejně jako u spojité Fourierovy transformace si definujeme inverzní (zpětnou) transformaci.

Definice 1.4 (**Inverzní diskrétní Fourierova transformace**)**.** Nechť $\mathbf{f}(k, l) : \{0, 1, \ldots, N-1\}^2 \to \mathbb{C}, N \in \mathbb{N}$ a nechť je funkce $F(\xi, \eta)$ její diskrétní Fourierovou transformací. Pak pro inverzní (zpětnou) diskrétní Fourierovu transformaci funkce $F(\xi, \eta)$ platí

$$\mathbf{f}(k, l) = \frac{1}{N} \sum_{\xi=0}^{N-1} \sum_{\eta=0}^{N-1} F(\xi, \eta) e^{-i\frac{2\pi}{N}(k\xi + l\eta)}$$

a značíme $\mathcal{D}^{-1}\{F(\xi, \eta)\}$.

Uveďme si nyní dvě základní vlastnosti diskrétní dvojrozměrné Fourierovy transformace z hlediska jejího využití při zpracování obrazu.

Střední hodnota Spektrální složka pro $\xi = 0$ a $\eta = 0$

$$F(0, 0) = \frac{1}{N} \sum_{\xi=0}^{N-1} \sum_{\eta=0}^{N-1} \mathbf{f}(k, l)$$

nám reprezentuje střední (průměrnou) hodnotu všech pixelů.

Periodicita Využitím substituce $\xi = \xi + mN$ a $\eta = \eta + nN$, kde $m, n \in \mathbb{Z}$, dostaneme

$$F(\xi + mN, \eta + nN) = \frac{1}{N} \sum_{\xi=0}^{N-1} \sum_{\eta=0}^{N-1} \mathbf{f}(k,l) e^{-i\frac{2\pi}{N}(k\xi+l\eta)} e^{-i2\pi(km+ln)},$$

odkud jde vidět, že druhý exponenciální výraz je roven jedné pro všechny $m, n \in \mathbb{Z}$ a tedy spektrum je periodické

$$F(\xi + mN, \eta + nN) = F(\xi, \eta), \quad m, n \in \mathbb{Z}.$$

1.5 Konvoluce a její vlastnosti

Definice 1.5. (**Konvoluce**) Nechť jsou definovány dvě reálné funkce $f(x,y)$ a $g(x,y)$, pak jejich konvolucí nazveme reálnou funkci dvou proměnných $h(x,y)$ definovanou jako integrál

$$h(x,y) = f(x,y) * g(x,y) = \iint_{\mathbb{R}^2} f(u,v) g(x-u, y-u) \mathrm{d}u\mathrm{d}v.$$

Funkci $g(x,y)$ nazýváme *konvoluční jádro*.

Konvoluce dále splňuje následující vlastnosti:

1. Komutativita

$$f * g = g * f, \quad \forall f(x,y), g(x,y) \in \mathbb{R}^2$$

2. Násobení konstantou

$$c_1 f * c_2 g = c_1 c_2 (f * g), \quad \forall f(x,y), g(x,y) \in \mathbb{R}^2, \forall c_1, c_2 \in \mathbb{R}$$

3. Distributivita vůči sčítání

$$f * (g_1 + g_2) = f * g_1 + f * g_2, \ \forall f(x,y), g_1(x,y), g_2(x,y) \in \mathbb{R}^2$$

4. Asociativita

$$f_1 * (f_2 * f_3) = (f_1 * f_2) * f_3, \quad \forall f_1(x,y), f_2(x,y), f_3(x,y) \in \mathbb{R}^2$$

Pro využití konvoluce ve zpracování obrazu jsou především důležité její vlastnosti spojené s Fourierovou transformací. Tyto vlastnosti jsou popsány tzv. *konvolučním teorémem.*

Věta 1.6 (**Konvoluční teorém**). *Nechť jsou dány dvě komplexní funkce $f(x,y)$, $g(x,y)$ definované na $\mathbb{R}^2$ a komplexní funkce $F(\xi,\eta)$, $G(\xi,\eta)$, které jsou jejich Fourierovými obrazy. Potom platí:*

1. $\mathcal{F}\{f(x,y) * g(x,y)\} = F(\xi,\eta)G(\xi,\eta)$

2. $\mathcal{F}\{f(x,y) \cdot g(x,y)\} = \frac{1}{4\pi^2} F(\xi,\eta) * G(\xi,\eta)$

Vlastnosti nám popisují vztah mezi operacemi násobení a konvoluce v prostorové a spektrální oblasti. Z těchto dvou vlastností je pro nás důležitá především první, která se používá při přechodu mezi spektrální a prostorovou oblastí. Příkladem může být filtrace užitím konvoluce, o které se budeme zmiňovat v dalším textu.

Diskrétní konvoluce

Definice 1.7 (**Diskrétní konvoluce**)**.** Uvažujme $\Omega = \{(i,j)|i = 0,1,\ldots,M-1;\ j = 0,1,\ldots,N-1\}$ a funkce $\mathbf{f}(i,j)$, $\mathbf{g}(i,j)$ z tohoto prostoru. Pak lze *diskrétní konvoluci* $\mathbf{h}(i,j)$ zapsat pomocí vztahů

$$\mathbf{h}(i,j) = (\mathbf{f} * \mathbf{g})(i,j) = \sum_{r=0}^{M-1}\sum_{s=0}^{N-1} \mathbf{f}(r,s)\mathbf{g}(i-r,j-s), \tag{1.3}$$

$$\mathbf{h}(i,j) = (\mathbf{g} * \mathbf{f})(i,j) = \sum_{r=0}^{M-1}\sum_{s=0}^{N-1} \mathbf{g}(r,s)\mathbf{f}(i-r,j-s). \tag{1.4}$$

Konvoluční jádro $\mathbf{g}(i,j)$ má běžně nenulové hodnoty pouze na velmi malém podintervalu svých hodnot. Je patrné, že při výpočtu konvoluce pomocí vztahů (1.3) nebo (1.4) je zapotřebí znát hodnoty funkcí $\mathbf{f}(i,j)$ a $\mathbf{g}(i,j)$ i mimo rozsah Ω. Tento problém můžeme vyřešit doplněním funkcí $\mathbf{f}(i,j)$ a $\mathbf{g}(i,j)$ i mimo Ω. Toho můžeme docílit například pomocí tzv. *lineární konvoluce*, u které předpokládáme, že funkce $\mathbf{f}(i,j)$ a $\mathbf{g}(i,j)$ nabývají mimo Ω hodnoty 0. Ačkoli nám tento přístup velice ovlivní výsledné hodnoty na okrajích obrazu, je tento přístup ve zpracování obrazu po většinou postačující a to z toho důvodu, že informace právě v okolí okrajů pro nás nejsou klíčové.

Uveďme si nicméně další možný přístup doplnění definičního oboru, který potlačuje okrajový efekt vznikající při lineární konvoluci. Tento přístup využívá zperiodičtění funkce $\mathbf{f}(i,j)$ resp. $\mathbf{g}(i,j)$, a to tak, že uděláme její symetrii nejprve s osami $i = 0$, $j = 0$, $i = M-1$, $j = N-1$ a poté středovou symetrii dle bodů $[0,0]$, $[0,N-1]$, $[M-1,0]$ a $[M-1,N-1]$. Právě tento přístup budeme uvažovat v dalším textu a při zpracovávání obrazové informace.

2 Zpracování obrazové informace

V této kapitole se budeme věnovat definicím pojmů souvisejících se zpracováním obrazu a metodám pro jeho zpracování. Informace k těmto účelů jsme primárně čerpali z [2][7][9][22].

2.1 Obrazová matice

Obraz můžeme obecně chápat jako spojitou funkci $f(x, y)$, kde $x, y \in \mathbb{R}$. Jednotlivé funkční hodnoty $f(x, y)$ nám můžou reprezentovat hodnoty jasu, barvy, intenzity světla atd. V dnešní době se však pracuje především s *digitálním obrazem*, který je reprezentován maticí o rozměrech $m \times n$, nazýváme ji obrazovou maticí a značíme $\mathcal{I}_{m\times n}$.

Abychom získali digitální obraz, je nutné provést tzv. *digitalizaci*. Tento proces se skládá ze dvou kroků, *vzorkování* (diskretizace v rovině) a *kvantizace* (diskretizace v úrovních).

Nejprve probíhá diskretizace v rovině, což znamená, že si rozdělíme definiční obor obrazové funkce $f(x, y)$ na konečné množství pravidelných obrazových elementů, které mají nejčastěji tvar čtverce či obdélníku a říkáme jim pixely[1]. Jejich poloha je určena sloupcovým indexem i a řádkovým indexem j. Pixel obrazové matice $\mathcal{I}_{m\times n}$ o souřadnicích $[i, j]$, kde $0 \leq i \leq m-1$, $0 \leq j \leq n-1$ budeme značit jako $\mathcal{P}[i, j]$.

V druhé fázi se ke každému takovému elementu obrazové matice $\mathcal{I}_{m\times n}$, nad kterým je obrazová funkce $f(x, y)$ definována, přiřadí reprezentační hodnota.

Stanovená reprezentační hodnota se odvíjí od bitové hloubky b, která nám udává počet úrovní, kterých mohou jednotlivé pixely nabývat. Obecně jsou tyto hodnoty dány jako

$$0 \leq \mathcal{P}[i, j] \leq 2^b.$$

Ve většině případů je postačující tzv. $8-$bitové kvantování obrazové informace, kdy dostáváme $2^8 = 256$ hodnot. V současné době je pro vědecké účely běžné použití 12-bitového (4096 úrovní) nebo dokonce 16-bitového kvantování (65536 úrovní). Pro zobrazení se výsledně zpracovaný obraz převádí na postačující 8-bitové kvantování.

Zaveďme si nyní pojem okolí pixelu $\mathcal{P}[i, j]$.

Definice 2.1 (**Kruhové okolí**)**.** Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i, j])$ a dále nechť máme konstantu $r > 0$. Pak množinu pixelů $\mathcal{Q}[k, l]$, pro kterou platí

$$\sqrt{(k-i)^2 + (l-j)^2} \leq r$$

nazveme *kruhovým okolím pixelu* $\mathcal{P}[i, j]$ a budeme ji značit $\mathcal{O}^P_\circ(i, j, r)$.

Definice 2.2 (**Obdélníkové okolí**)**.** Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i, j])$ a dále nechť máme konstantu $r > 0$ a $s > 0$. Pak množinu pixelů $\mathcal{Q}[k, l]$, pro kterou platí

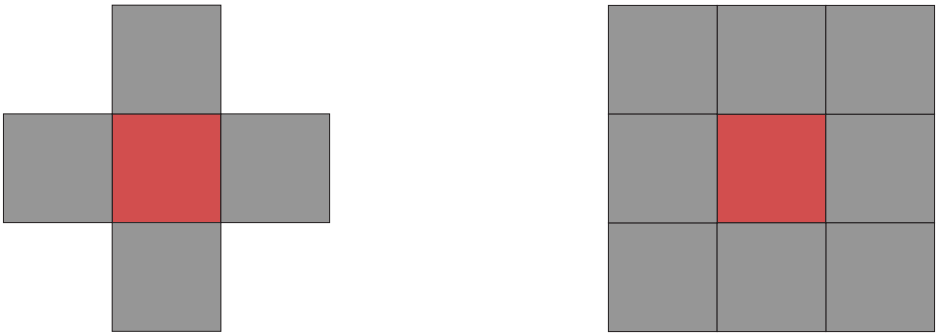
$$k, l \in \langle i-r, i+r\rangle \times \langle j-s, j+s\rangle$$

nazveme *obdélníkovým okolím pixelu* $\mathcal{P}[i, j]$ a budeme ji značit $\mathcal{O}^P_\square(i, j, r, s)$.

V případě, že jsou si konstanty r a s rovny, bavíme se o čtvercovém okolí pixelu $\mathcal{P}[i, j]$. Kromě těchto okolí se používají i jejich podmnožiny. Těmi nejčastějšími typy jsou tzv. *čtyřsousednost* $\mathcal{N}^P_4$ a *osmisousednost* $\mathcal{N}^P_8$, které můžeme vidět na obrázku (1).

[1] z anglického picture element

Definice 2.3 (**Spojitelnost**). Nechť máme dva pixely $\mathcal{P}[i,j]$ a $\mathcal{Q}[k,l]$ z obrazové matice $\mathcal{I}$. Pak řekneme, že pixely $\mathcal{P}[i,j]$ a $\mathcal{Q}[k,l]$ jsou *4-spojitelné*, resp. *8-spojitelné* jestliže $\mathcal{Q}[k,l] \in \mathcal{N}_4^P$, resp. $\mathcal{Q}[k,l] \in \mathcal{N}_8^P$. Pixely $\mathcal{P}[i,j]$ a $\mathcal{Q}[k,l]$ jsou *spojitelné*, pokud jsou 4-spojitelné nebo 8-spojitelné.



(a) Čtyřsousednost (b) Osmisousednost

Obrázek 1: Znázornění dvou druhů okolí pixelu, který je zobrazen červeně

2.1.1 Druhy digitálního obrazu

V zásadě můžeme rozlišovat tři základní druhy digitálního obrazu a to:

- Monochromatický (šedotónový) obraz
- Multispektrální (barevný) obraz
- Binární obraz

Monochromatický obraz odpovídá případu, který jsme si popsali výše, to znamená, že každý pixel $\mathcal{P}[i,j]$ obrazu má jedinou hodnotu, která je vybrána z rozmezí pro danou bitovou hloubku a říká se jí odstín šedi.

U *multispektrálního obrazu* jsou jednotlivé pixely $\mathcal{P}[i,j]$ reprezentovány jako vektory, jejichž hodnoty odpovídají jasu v jednotlivých barevných složkách, tj. červené, zelené a modré (tzv. *RGB model*). Při digitalizaci barevného obrazu probíhá zpracování pro každou z těchto složek zvlášť.

Poslední typ, *binární obraz*, je reprezentován obrazovou maticí, jejíž prvky nabývají pouze dvou hodnot a to 0 a 1, které odpovídají černé a bílé.

V následujících kapitolách se omezíme výhradně na monochromatické obrazy a to především z důvodu jednodušší ukázky aplikace jednotlivých metod a algoritmů. Nicméně většinu procesů lze aplikovat i na multispektrální obraz, a to použitím na jednotlivé složky.

2.1.2 Histogram

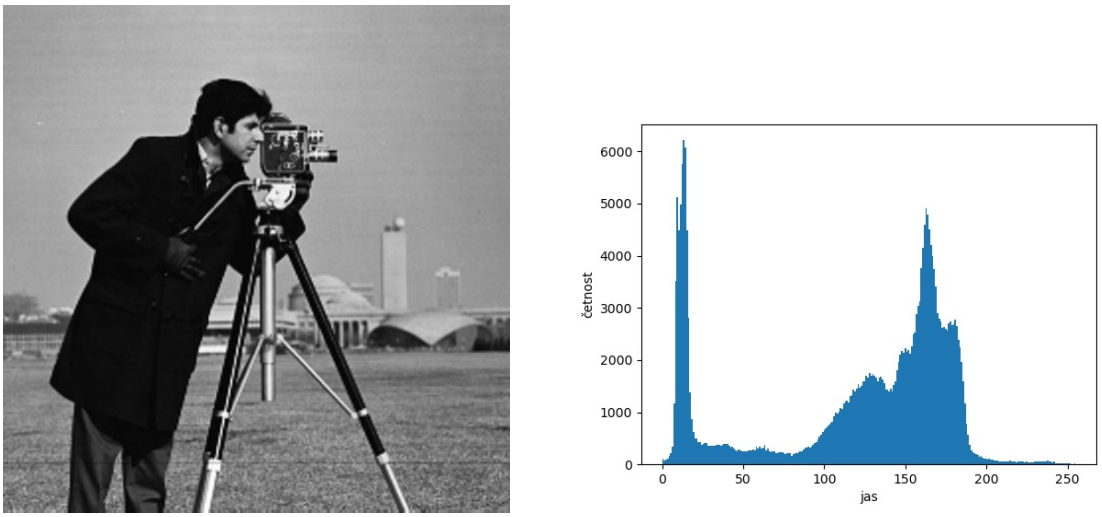
Zaveďme si nyní velice důležitý zdroj informací o poměru jasu a kontrastu v obraze.

Definice 2.4 (**Histogram**). Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i,j])$. Dále nechť nám hodnota $2^b - 1$, kde b určuje bitovou hloubku, udává počet úrovní, kterých může pixel $\mathcal{P}[i,j]$ nabývat. Pak *histogramem* H rozumíme funkci H, která je dána jako

$$H(z) = N_z,$$

kde $z \in \langle 0, 2^b - 1 \rangle$ a $N_z = \sum_{\forall \mathcal{P}[i,j]=z} 1$.

Histogram nám tedy ukazuje využití dynamického rozsahu a četnosti jednotlivých úrovní jasu.



(a) Obraz (b) Histogram obrazu

Obrázek 2: Ukázka monochromatického, 8−bitového obrazu a jeho histogramu

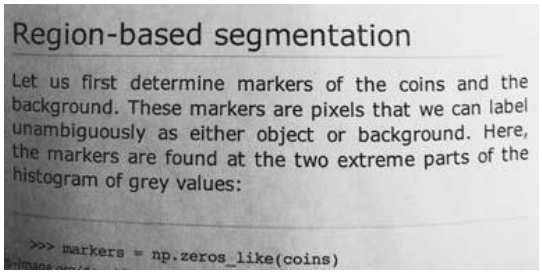
2.2 Ekvalizace histogramu

V mnoha případech nastává situace, kdy jsou hodnoty v histogramu soustředěny pouze v určitém malém rozmezí dynamického rozsahu, což nám signalizuje nízký kontrast v obraze. Z tohoto důvodů se provádí tzv. *ekvalizace histogramu*. Nejjednodušším příkladem takovéto úpravy je pomocí *roztažení histogramu*.

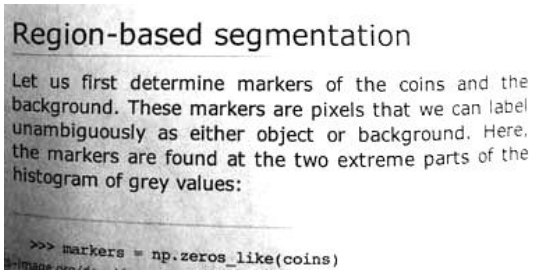
Uvažujme obrazovou matici $\mathcal{I} = (\mathcal{P}[i,j])$ s nízkým kontrastem. Hodnoty pixelů $\mathcal{P}[i,j]$ jsou v rozmezí $\mathcal{P}_{min}$ a $\mathcal{P}_{max}$, kde tyto hodnoty nám označují minimální, resp. maximální hodnotu pixelu $\mathcal{P}[i,j]$ v obrazové matici $\mathcal{I}$. Abychom zvýšili kontrast takového obrazu, musíme hodnoty roztáhnout na větší rozpětí hodnot $[z_1, z_k]$. Hodnoty obrazové matice $\mathcal{I}^{new} = (\mathcal{Q}[i,j])$ dostaneme pomocí vztahu

$$\mathcal{Q}[i,j] = \frac{\mathcal{P}[i,j] - \mathcal{P}_{min}}{\mathcal{P}_{max} - \mathcal{P}_{min}}(z_k - z_1) + z_1 \quad \forall \mathcal{P}[i,j] \in \mathcal{I}.$$

Na obrázku (3) může vidět ukázku použití ekvalizace histogramu. Úpravy histogramu se obecně provádějí pouze pro vizualizaci dat, s takto upravenými obrazy dále nepracujeme. Informace k této části jsme čerpali z [1][2][9].



(a) Originální obraz



(b) Obraz po ekvalizaci histogramu

Obrázek 3: Ukázka použití ekvalizace histogramu

2.3 Šum

Šum je nevyhnutelný vedlejší efekt pořizování obrazu a i v případě, že není šum v obraze vidět, vždy existuje. Toto je způsobeno povahou elektronických součástek, které přijímají a vysílají signál s šumem. V obrazech ho můžeme pozorovat jako zrnitost a jedná se o odchylku intenzity jednotlivých pixelů obrazové matice od jejich skutečných hodnot. V obrazech se nejčastěji vyskytuje šum, který má tzv. *aditivní model*. Informace k této části jsme čerpali zejména z [11][12][17].

2.3.1 Aditivní model šumu

Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i,j])$, která neobsahuje šum. Pak obrazovou matici $\mathcal{I}^{new} = (\mathcal{Q}[i,j])$, která vznikne degradací matice $\mathcal{I}$ šumem, který se řídí aditivním modelem, dostaneme podle vztahu

$$\mathcal{Q}[i,j] = \mathcal{P}[i,j] + n[i,j],$$

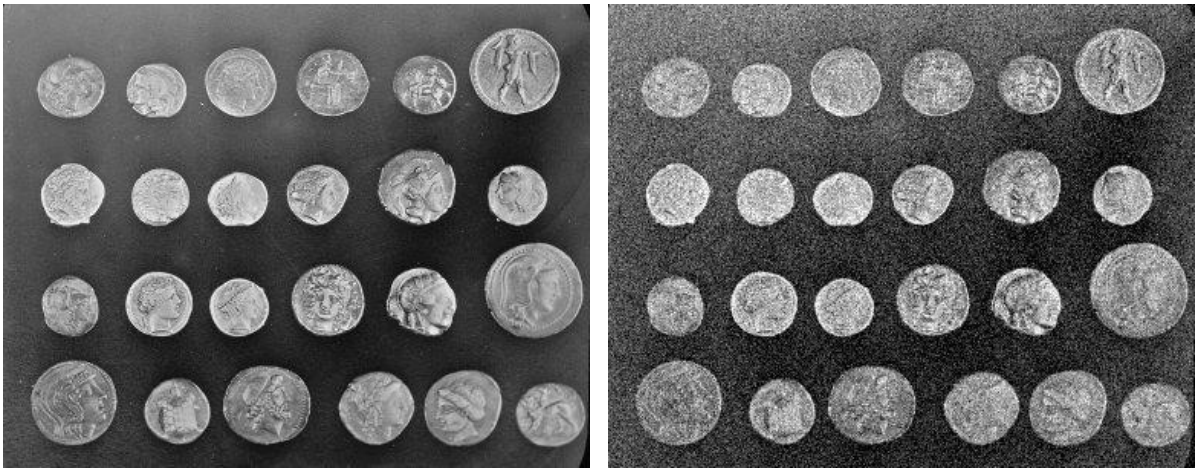
kde $n[i,j]$ nám označuje šum. Aditivní šum může mít různá rozložení, tím nejběžnějším je ale Gaussovské.

2.3.2 Gaussův šum

Gaussův šum je jedním z nejčastěji vyskytujících se šumů v obrazech vůbec. Je to klasický příklad aditivního šumu, kdy je k pravé hodnotě pixelu přičítána náhodná hodnota, kterou můžeme popsat pomocí vztahu

$$p_G(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}},$$

kde z reprezentuje úroveň jasu, μ střední hodnotu hodnot jasu a σ představuje její směrodatnou odchylku. Ukázku obrazu s Gaussovským šumem můžeme vidět na obrázku (4).



(a) Originální obraz (b) Obraz degradovaný Gaussovým šumem

Obrázek 4: Ukázka Gaussova šumu

2.4 Lineární filtrace

Informace k této kapitole jsme čerpali převážně z [2][5][12][17][22]. Lineární filtry mají širokou škálu využitelnosti. Jak jsme již zmiňovali v kapitole (1.5), lineární filtrace nám upravuje frekvenční spektrum zpracovávaného obrazu. Lineární filtry mají tedy základ v konvoluci a právě volba konvolučního jádra může velmi ovlivnit jejich vlastnosti.

Definice 2.5 (**Lineární filtr**)**.** Nechť máme obrazové matice $\mathcal{I}^{in} = (\mathcal{P}[i,j])$, $\mathcal{I}^{new} = (\mathcal{Q}[i,j])$. Dále nechť $\mathbf{H} = (h[k,l])$ je konvoluční jádro o rozměrech $2m+1 \times 2n+1$, kde $0 \leq k \ \leq 2m$ a $0 \leq l \leq 2n$. Pak zobrazení $F_{lin} : \mathcal{P}[i,j] \to \mathcal{Q}[i,j]$, pro které platí:

$$\mathcal{Q}[i,j] = \sum_{k=-m}^{m} \sum_{l=-n}^{n} \mathcal{P}[i-k, j-l] h[k+m, l+n]$$

nazveme *lineárním filtrem* s konvolučním jádrem $\mathbf{H}$.

Podívejme se nyní na několik základních příkladů lineárních filtrů, které budeme dále využívat.

Filtr typu dolní propust Tento typ filtrace má za účel propouštět pouze nízké frekvence signálu a potlačovat frekvence vysoké, mezi které patří například frekvence šumu. Jeho nejjednodušším příkladem je tzv. klouzavý průměr, jehož konvoluční jádro $\mathbf{H}_P$ o velikosti 3×3 má podobu

$$\mathbf{H}_P = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Tento filtr nám bohužel v některých případech propouští i určité množství vyšších frekvencí, což samozřejmě není žádoucí. Tento nedostatek nám vyřeší použití následujícího filtru.

Gaussův filtr Prvky tohoto filtru jsou dány Gaussovou funkcí dvou proměnných, která má obecně pro diskrétní souřadnice tvar

$$g(i,j) = A\, e^{\left[-\left(\frac{(i^2-\mu_i)}{2\sigma_i^2}+\frac{(j^2-\mu_j)}{2\sigma_j^2}\right)\right]},$$

kde $A, \sigma_i, \sigma_j > 0$ a $\mu_i, \mu_j \in \mathbb{R}$. V našem případě budeme uvažovat $\mu_i = \mu_j = 0$, $\sigma_i = \sigma_j = \sigma$ a A volíme tak, aby součet všech prvků konvolučního jádra $\mathbf{H}_G$ byl roven 1. Po úpravě tedy dostaneme tvar

$$g(i,j) = Ae^{-\frac{(i^2+j^2)}{2\sigma^2}}. \tag{2.1}$$

Pro jádro velikosti 7×7 a $\sigma = 2$ dostaneme konvoluční jádro, které má podobu

$$\mathbf{H}_G = \frac{1}{21.4125}\begin{bmatrix} 0.105 & 0.197 & 0.287 & 0.325 & 0.287 & 0.197 & 0.105 \\ 0.197 & 0.368 & 0.535 & 0.607 & 0.535 & 0.38 & 0.197 \\ 0.287 & 0.535 & 0.779 & 0.882 & 0.779 & 0.535 & 0.287 \\ 0.325 & 0.607 & 0.882 & 1 & 0.882 & 0.607 & 0.325 \\ 0.287 & 0.535 & 0.779 & 0.882 & 0.779 & 0.535 & 0.287 \\ 0.197 & 0.368 & 0.535 & 0.607 & 0.535 & 0.38 & 0.197 \\ 0.105 & 0.197 & 0.287 & 0.325 & 0.287 & 0.197 & 0.105 \end{bmatrix}.$$

Pro urychlení výpočtu, je možné toto jádro rozložit na řádkovou a sloupcovou matici, a to díky symetrii této matice. Konvoluci pak postupně provedeme s oběma těmito rozloženými maticemi. Tento postup je možný díky asociativitě konvoluce.

2.5 Nelineární filtrace

Obecně při aplikaci filtrů přiřazujeme vyšetřovanému pixelu $\mathcal{P}[i,j]$ hodnotu, která je důsledkem nějaké operace mezi pixely z jeho okolí. Pokud je tato operaci jiná, než lineární kombinace, bavíme se o tzv. *nelineárních filtrech*. Podobně jako u lineárních filtrů, jejichž vlastnosti závisí na volbě konvoluční jádra, u těchto filtrů závisí vlastnosti na volbě okolí $\mathcal{N}^{\mathcal{P}}(i,j)$ vyšetřovaného pixelu $\mathcal{P}[i,j]$. Informace k této kapitole jsme čerpali z [1][14][15][22].

Filtr typu maxima a minima Nejjednodušším případem nelineárních filtrů jsou filtry typu maxima a minima.

Definice 2.6 (**Filtr typu maximum**)**.** Nechť máme obrazové matice $\mathcal{I}^{in} = (\mathcal{P}[i,j])$, $\mathcal{I}^{new} = (\mathcal{Q}[i,j])$. Dále nechť je dáno okolí $\mathcal{O}^{\mathcal{P}}(i,j)$ pixelu $\mathcal{P}[i,j]$. Pak zobrazení F_{max} : $\mathcal{P}[i,j] \to \mathcal{Q}[i,j]$, které dáno jako

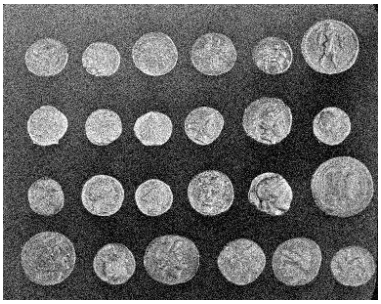
$$\mathcal{Q}[i,j] = \max \mathcal{O}^{\mathcal{P}}(i,j)$$

nazveme *filtrem typu maximum*.

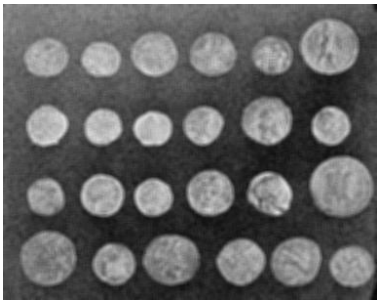
Definice 2.7 (**Filtr typu minimum**)**.** Nechť máme obrazové matice $\mathcal{I}^{in} = (\mathcal{P}[i,j])$, $\mathcal{I}^{new} = (\mathcal{Q}[i,j])$. Dále nechť je dáno okolí $\mathcal{O}^{\mathcal{P}}(i,j)$ pixelu $\mathcal{P}[i,j]$. Pak zobrazení $F_{min} : \mathcal{P}[i,j] \to \mathcal{Q}[i,j]$, které dáno jako

$$\mathcal{Q}[i,j] = \min \mathcal{O}^{\mathcal{P}}(i,j)$$

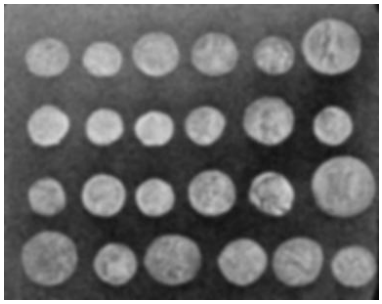
nazveme *filtrem typu minima.*



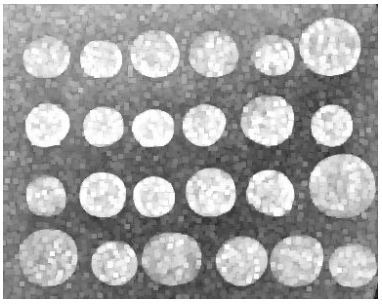
(a) Degradovaný obraz Gaussovým šumem



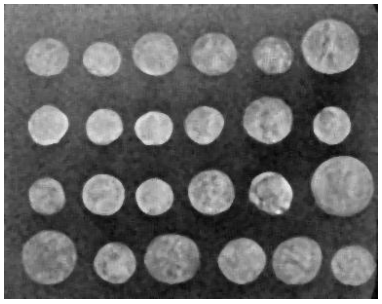
(b) Filtr typu dolní propust s velikostí jádra 5×5



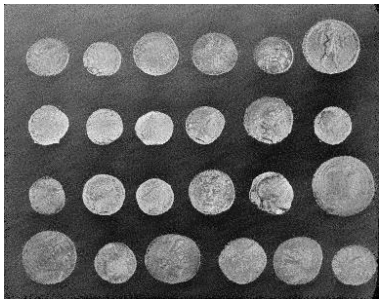
(c) Gaussův filtr s parametrem $\sigma = 2$



(d) Filtr typu maxima s velikostí okolí 5×5



(e) Mediánový filtr s velikostí okolí 5×5



(f) Bilaterální filtr s parametry $\sigma_s = 2$ a $\sigma_r = 0.11$

Obrázek 5: Srovnání výsledků po aplikaci filtrů na zašuměný obraz

Mediánový filtr Pro nás nejdůležitějším a velmi často používaným filtrem je filtr mediánový.

Definice 2.8 (**Mediánový filtr**)**.** Nechť máme obrazové matice $\mathcal{I}^{in} = (\mathcal{P}[i,j])$ a $\mathcal{I}^{new} = (\mathcal{Q}[i,j])$. Dále nechť je dáno okolí $\mathcal{O}^{\mathcal{P}}(i,j)$ pixelu $\mathcal{P}[i,j]$. Pak zobrazení $F_{med} : \mathcal{P}[i,j] \to \mathcal{Q}[i,j]$, které dáno jako

$$\mathcal{Q}[i,j] = \text{median } \mathcal{O}^{\mathcal{P}}(i,j)$$

nazveme *mediánovým filtrem.*

Bilaterální filtr Posledním příkladem nelineárních filtrů je tzv. *bilaterální filtr*. Tento filtr využívá, podobně jako lineární filtry, konvoluci s obrazovou maticí. Tato maska je nicméně proměnná v závislosti na okolí vyšetřovaného pixelu.

Definice 2.9 (**Bilaterální filtr**)**.** Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i,j])$. A pixely $\mathcal{Q}[k,l] \subset \mathcal{I}$, které jsou součástí konvolučního jádra $\mathbf{H}_{BF}$. *Bilaterální filtr* $BF_{\mathcal{P}}$ pro vyšetřovaný pixel $\mathcal{P}[i,j]$ je pak dán jako

$$BF_{\mathcal{P}} = \frac{1}{k_{\mathcal{P}}} \sum_{\mathcal{Q} \in \mathbf{H}} g_{\sigma_s}(||\mathcal{P} - \mathcal{Q}||) g_{\sigma_r}(|\mathcal{I}_{\mathcal{P}} - \mathcal{I}_{\mathcal{Q}}|) \mathcal{I}_{\mathcal{Q}},$$

kde zápisem $\mathcal{I}_{\mathcal{P}}$, resp. $\mathcal{I}_{\mathcal{Q}}$ rozumíme intenzitu pixelu $\mathcal{P}$, resp. $\mathcal{Q}$ a g_{σ_s}, resp. g_{σ_r} jsou Gaussovy funkce s parametry σ_s, resp. σ_r. Koeficient normalizace $k_{\mathcal{P}}$ je dán jako

$$k_{\mathcal{P}} = \sum_{\mathcal{Q} \in \mathbf{H}} g_{\sigma_s}(||\mathcal{P} - \mathcal{Q}||) g_{\sigma_r}(|\mathcal{I}_{\mathcal{P}} - \mathcal{I}_{\mathcal{Q}}|).$$

Oproti Gaussovu filtru, kterému odpovídá část $g_{\sigma_s}(||\mathcal{P} - \mathcal{Q}||)$, a která nám přiřazuje váhu pixelu v závislosti na jeho vzdálenosti od středu konvolučního jádra $\mathbf{H}_{BF}$, zde využíváme navíc část $g_{\sigma_r}(|\mathcal{I}_{\mathcal{P}} - \mathcal{I}_{\mathcal{Q}}|)$, která nám určuje váhu pixelů v závislosti na rozdílu jeho intenzity od intenzity středu konvolučního jádra $\mathbf{H}_{BF}$.

Tento filtr řeší nedostatek Gaussova filtru, který rozmazává obraz a dochází tak ke ztrátě informací. Porovnání výsledků po aplikaci různých filtrů na obraz s Gaussovým šumem můžeme vidět na obrázku (5), speciálně pak můžeme vidět porovnání použití Gaussova filtru a bilaterálního filtru na obrázcích (5c) a (5f). Nevýhodou bilaterálního filtru je jeho výpočetní náročnost, která je způsobena přepočítáváním konvolučního jádra, pro každý pixel.

2.6 Segmentace obrazu

V praxi často nastává situace, kdy nás zajímají pouze některé části obrazu, které mají podobné charakteristiky. K těmto účelům nám slouží *segmentace obrazu*, která je velice důležitým procesem při zpracování obrazu. Hlavním úkolem segmentace je zjednodušení, tj. znázornění obrazu v podobě, která je lépe analyzovatelná. Tedy cílem segmentace je rozdělení obrazové matice $\mathcal{I} = (\mathcal{P}[i,j])$ na několik objektů $O_1, \ldots, O_k$, které mají podobné vlastnosti a znaky. Jednotlivé objekty pak musí splňovat následující definice.

Definice 2.10. Objekt O_i je podmnožinou obrazu.

Definice 2.11. Segmentací rozdělujeme pixely obrazu na objekty, takové že

- $\bigcup_{i=1}^{k} O_i = \mathcal{I}$
- $O_i \bigcap O_j = \emptyset$ pro $\forall i \neq j$

V následujícím textu si uvedeme několik různých metod segmentace obrazu. Informace k této části jsme čerpali z [1][2][3][10][13][16][18][20][21].

2.6.1 Prahování

Tento typ segmentace patří k nejjednodušším technikám, ale zároveň k nejpoužívanějším. Její princip spočívá v rozdělení pixelů obrazu podle intenzity pixelů obrazové matice $\mathcal{I} = (\mathcal{P}[i,j])$. Nejběžněji využíváme prahování pro rozdělení obrazové matice $\mathcal{I}$ na dvě

části, objekt a pozadí, v závislosti na hodnotě prahu τ. Obrazovou matici $\mathcal{I}^{new} = \mathcal{Q}[i,j]$, kterou získáme prahováním obrazové matice $\mathcal{I}$, pak máme danou jako

$$\mathcal{I}^{new} = \begin{cases} 1 & \text{pro} \quad \mathcal{P}[i,j] > \tau \\ 0 & \text{pro} \quad \mathcal{P}[i,j] \leq \tau \end{cases},$$

kde hodnota 1 nám značí pixely objektu a 0 pixely pozadí. Tím nám tedy vznikne binární obraz, ve kterém máme bílé objekty na černém pozadí. Dále můžeme prahování rozdělit do několika kategorií podle množství a podoby prahu τ.

- Globální prahování - Práh τ je konstantní pro celý obraz.
- Lokální prahování - Také se mu někdy říká regionální, obraz je rozdělen na několik oblastí, kdy pro každou z nich je určen lokální práh.
- Adaptivní prahování - Práh τ je závislý na souřadnicích $[i,j]$.
- Vícestupňové prahování - V obraze existuje víc než jeden práh tj. pro dva prahy máme

$$\mathcal{I}^{new} = \begin{cases} a & \text{pro} \quad \mathcal{P}[i,j] > \tau_2 \\ b & \text{pro} \quad \tau_1 < \mathcal{P}[i,j] \leq \tau_2 \\ c & \text{pro} \quad \mathcal{P}[i,j] \leq \tau_1 \end{cases}.$$

Nutno podotknout, že segmentace, u které bychom vyžadovali nalezení více než dvou prahů, je velice náročná a v takové situaci se preferuje použití například lokálního prahování, případně některé z metod, které budou popsány v dalším textu.

Prahování může být velice účinná metoda, především pro její jednoduchost, nicméně pro její správné fungování je zapotřebí, aby bylo splněno několik podmínek. Tou první je viditelné rozlišení mezi vrcholy histogramu, tzn. dostatečný kontrast mezi objektem a pozadím v obraze. Další problémem pro využití prahování může způsobit velké množství šumu, jenž zapříčiní překrytí vrcholů histogramu a tím pádem není možné zvolit vhodnou hodnotu prahu. V neposlední řadě může být problémem velký rozdíl ve velikostech objektu v poměru k velikosti pozadí a tedy nerozeznatelné vrcholy histogramu. Na obrázku (6) můžeme vidět ukázky histogramů a zvolenými prahy.

Pro hledání prahu existuje několik různých přístupů, uveďme si nyní dva základní algoritmy, pro jeho nalezení.

Klasická metoda hledání globálního prahu Tato metoda je základním algoritmem při hledání prahu obrazové matice $\mathcal{I} = (\mathcal{P}[i,j])$ a probíhá v několika krocích na základě stanoveného počátečního prahu τ.

1. V prvním kroku si zvolíme libovolnou hodnotu počátečního prahu τ.
2. Tím se náš obraz rozdělí na dvě skupiny G_1 a G_2 v závislosti na τ

$$\begin{aligned} G_1 \quad & \text{pro } \mathcal{P}[i,j] > \tau, \\ G_2 \quad & \text{pro } \mathcal{P}[i,j] \leq \tau. \end{aligned}$$

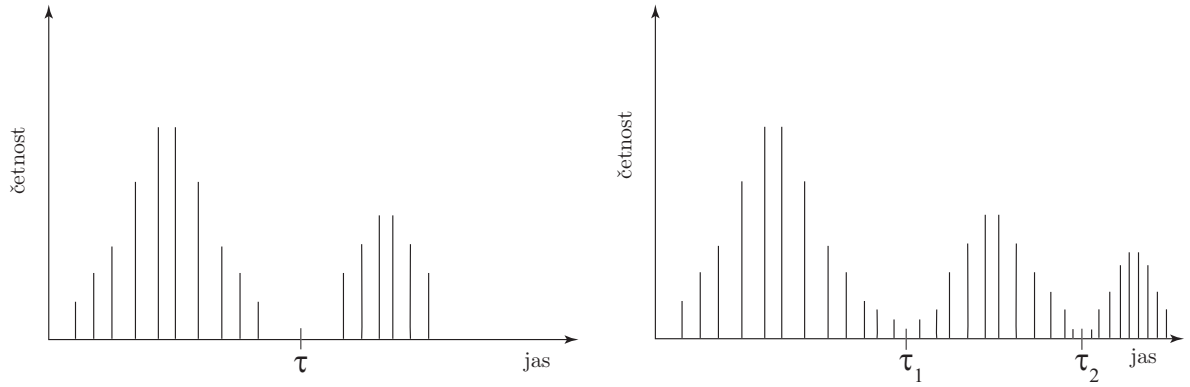
3. Pro každou skupinu vypočítáme průměrnou intenzitu m_1 a m_2 šedých hodnot.

4. Nyní určíme novou hodnotu prahu na základě průměrných intenzit m_1 a m_2 jako

$$\tau = \frac{1}{2}(m_1 + m_2).$$

5. Takto opakujeme kroky (2) až (4) do té doby, než bude rozdíl dvou po sobě vypočítaných prahů menší než stanovená hodnota $\triangle\tau$.

Jako počáteční hodnotu prahu můžeme volit například průměrnou hodnotu intenzity obrazové matice $\mathcal{I}$.



(a) Histogram se dvěma vrcholy a jedním prahem

(b) Histogram se třemi vrcholy a dvěma prahy

Obrázek 6: Ukázka histogramů pro jeden a více hodnot prahů

Otsuova metoda Druhá metoda, která je jedním z nejběžněji využívaných algoritmů hledání prahu, je *Otsuova metoda*. Tato metoda pracuje na principu diskriminace dvou tříd (kopce histogramu) a snaží se tak najít optimální hodnotu prahu τ. Nalezený práh minimalizuje vážený rozptyl σ_w dvou tříd jasů, tj. třídy jasů pozadí b a popředí f, definovaný jako vážený součet rozptylů těchto dvou tříd, tedy

$$\sigma_w^2(t) = w_b(t)\sigma_b^2(t) + w_f(t)\sigma_f^2(t),$$

kde váhy

$$w_b(t) = \sum_{i=1}^{t} p(i),$$
$$w_f(t) = \sum_{i=t+1}^{I} p(i)$$

jsou pravděpodobnosti dvou tříd rozdělených prahem τ. Střední hodnota těchto tříd je pak určena jako

$$\mu_b(t) = \frac{1}{w_b(t)} \sum_{i=1}^{t} ip(i),$$

$$\mu_f(t) = \frac{1}{w_f(t)} \sum_{i=t+1}^{I} ip(i)$$

a jejich rozptyly

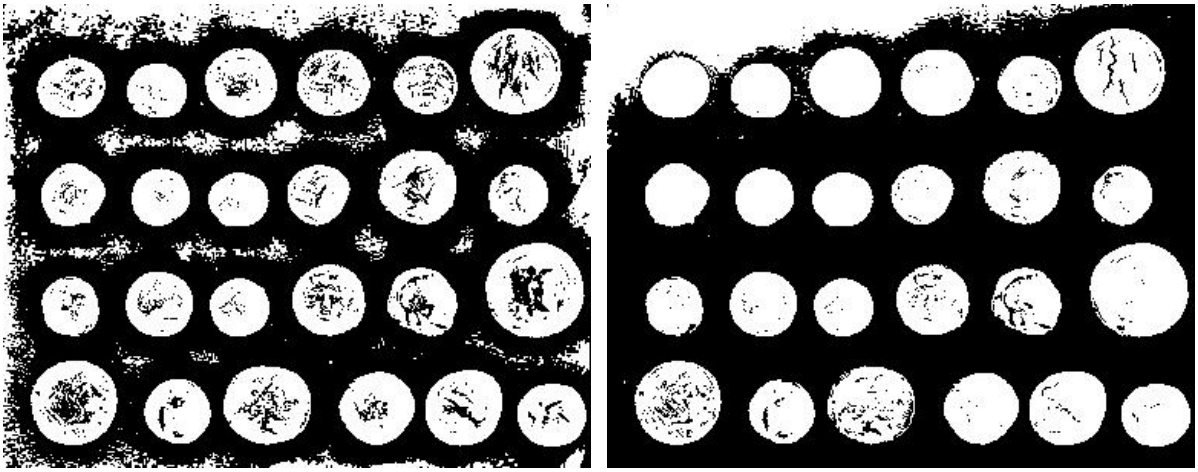
$$\sigma_b^2(t) = \frac{1}{w_b} \sum_{i=1}^{t} [i - \mu_b(t)]^2 p(i),$$
$$\sigma_f^2(t) = \frac{1}{w_f} \sum_{i=t+1}^{I} [i - \mu_f(t)]^2 p(i).$$

Tento přístup, tedy minimalizace rozptylu uvnitř tříd je ekvivalentní maximalizování rozptylu mezi třídami, který je dán vztahem

$$\begin{aligned} \sigma_b^2 &= \sigma^2 - \sigma_w^2 \\ &= w_b \left(\mu_b - \mu\right)^2 + w_f \left(\mu_f - \mu\right)^2 \\ &= w_b w_f \left(\mu_b - \mu_f\right)^2, \end{aligned}$$

kde $\mu = w_b\mu_b + w_f\mu_f$. Tento přístup je při výpočtu znatelně rychlejší. Otsuova metoda může být dále rozšířena i pro výpočet více prahů, více je možné nalézt například v [19].

Na obrázku (7) můžeme vidět srovnání výsledků při použití dvou zmíněných metod hledání prahu v obraze (4a).



(a) Výsledný obraz po použití základní metody globálního prahování

(b) Výsledný obraz po použití Otsuovy metody

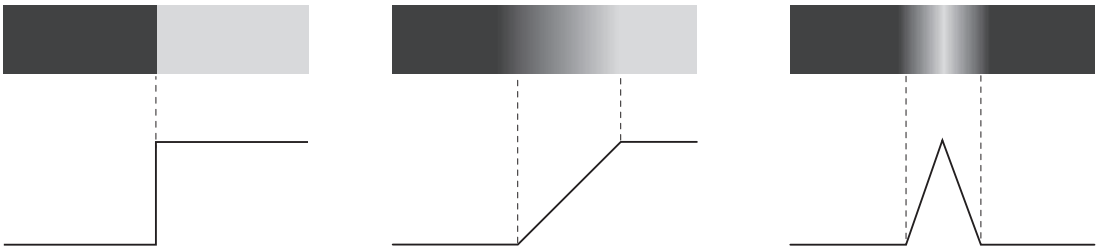
Obrázek 7: Srovnání základní metody globálního prahu a Otsuovy metody

2.6.2 Detekce hran

Metody segmentace na základě hledání hran jsou založeny na vlastnosti hrany, u které dochází k prudké změně hodnot intenzity pixelů obrazu. Ideální hrana by měla mít tloušťku jeden pixel, což se ale v praxi nestává z důvodu rozmazání nebo kvůli šumu. Na obrázku (8) můžeme vidět příklady jasových profilů v okolí hranových bodů. Metody, hledající hrany v obraze můžeme rozdělit do dvou základních skupin, a to na metody využívající první, resp. druhou derivaci.

Nutno podotknout, že před použitím těchto metod je zapotřebí provést filtraci šumu v obraze, např. Gaussovým filtrem, jelikož derivace, na kterých jsou tyto metody založeny, jsou na šum velice citlivé.

Nejprve se budeme věnovat metodám využívajícím první derivaci, kterým se také někdy říká *gradientní*, jelikož jsou založeny na hledání míst v obraze s největší změnou intenzity, tj. největším gradientem. Gradientem rozumíme vektor, který je kolmý na směrový vektor hrany. K nalezení hran v obrazové matici $\mathcal{I}$ se používají tzv. *hranové operátory*, které jsou dány příslušnými konvolučními jádry. Jedná se tedy o další typy lineárních filtrů.



(a) Skoková hrana (b) Šikmá hrana (c) Střechová hrana

Obrázek 8: Ukázka idealizovaných jasových profilů v okolí hranových bodů

Sobelův operátor *Sobelův operátor* používá dvě konvoluční jádra velikosti 3×3, pomocí kterých se snaží o nalezení gradientu v horizontálním směru G_x a vertikálním směru G_y. Konvoluční jádro $\mathbf{H}_x$ používané pro nalezení gradientu v horizontálním směru má podobu

$$\mathbf{H}_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

a pro vertikální směr pak

$$\mathbf{H}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}.$$

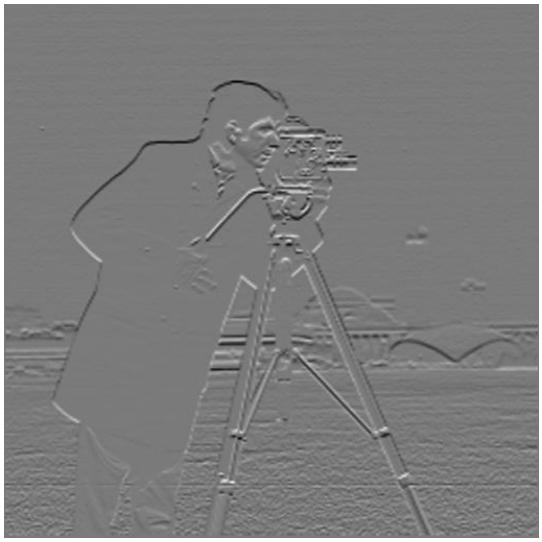
Tyto jádra jsou postupně pomocí konvoluce aplikovány na obrazovou matici $\mathcal{I}$, čímž nám vznikne tzv. *gradientní obraz*. Pomocí gradientů v jednotlivých směrech G_x a G_y pak vypočteme výslednou velikost gradientu G podle vztahu

$$G = \sqrt{G_x^2 + G_y^2}$$

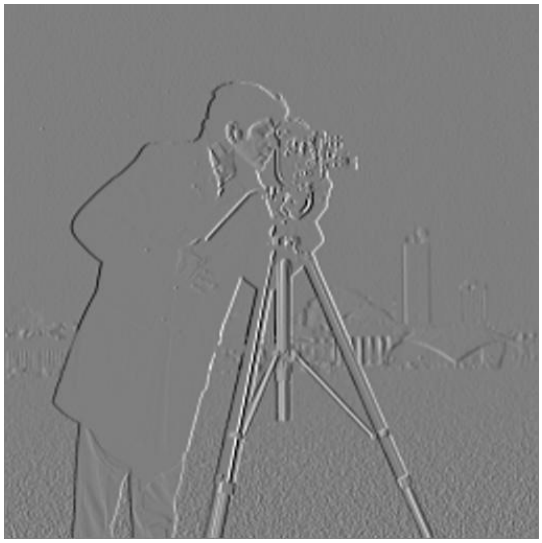
a úhel, který nám svírá gradient s osou x je dán jako

$$\theta = \arctan\left(\frac{G_x}{G_y}\right).$$

(a) Originální obraz



(b) Sobelův operátor v horizontálním směru



(c) Sobelův operátor ve vertikálním směru

(d) Výsledné hrany

Obrázek 9: Ukázka použití Sobelova operátoru

Pokud je úhel $\theta = 0$ tak to znamená, že má obraz vertikální hranu, jejíž levá část je tmavší než pravá. Na obrázku (9) můžeme vidět použití Sobelova operátoru.

Sobelův operátor je jedním z nejčastěji používaných hranových operátorů využívajících první derivace, ale není jediným. Dalšími příklady pak jsou:

- Robertsův operátor

$$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

- Prewittův operátor

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

- Frei-Chenův operátor

$$\begin{bmatrix} -1 & 0 & 1 \\ -\sqrt{2} & 0 & \sqrt{2} \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ 1 & \sqrt{2} & 1 \end{bmatrix}$$

- Kirschův operátor

$$\begin{bmatrix} -5 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & 3 & 3 \end{bmatrix} \quad \begin{bmatrix} 3 & 3 & 3 \\ 3 & 0 & 3 \\ -5 & -5 & -5 \end{bmatrix}.$$

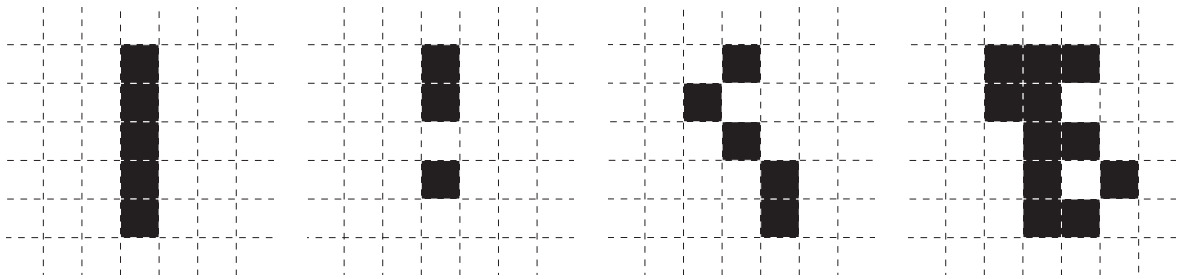
U této metody, tedy využití první derivace k nalezení hrany, dochází k problému, kdy jsou nalezené hrany příliš tlusté, tedy přesná hrana je těžko identifikovatelná. Z tohoto důvodu přišel J.F.Canny s myšlenkou „ideálního“ detektoru hran, jehož princip si nyní přiblížíme.

Cannyho detektor hran Tento algoritmus kromě detekce hran v obraze redukuje i jeho šum. Jeho průběh můžeme popsat v několika krocích, které se snaží o co nejlepší detekci hran, tak aby vyhovovaly následujícím podmínkám detekce.:

- Dobrá detekce - Optimální detektor musí minimalizovat pravděpodobnost detekce hran způsobených šumem, stejně tak jako nedetekování reálných hran.

- Dobrá lokalizace - Nalezené hrany musí být to co nejblíže reálným hranám.

- Jednoznačnost - Detektor musí vrátit pouze jednu odezvu pro každou hranu, nesmí dojít ke zdvojení.



(a) Skutečná hrana (b) Nespolehlivě detekovaná hrana (c) Špatně lokalizovaná hrana (d) Nejednoznačně detekovaná hrana

Obrázek 10: Podmínky Cannyho hranového detektoru

Vizualizované podmínky Cannyho hranového detektoru můžeme vidět na obrázku (10). Uvažujme obrazovou matici $\mathcal{I} = (\mathcal{P}[i, j])$, ve které budeme hrany hledat. Samotný proces detekce hran probíhá v pěti krocích a to postupně:

1. Aplikace vhodného Gaussova filtru na obrazovou matici $\mathcal{I}$.

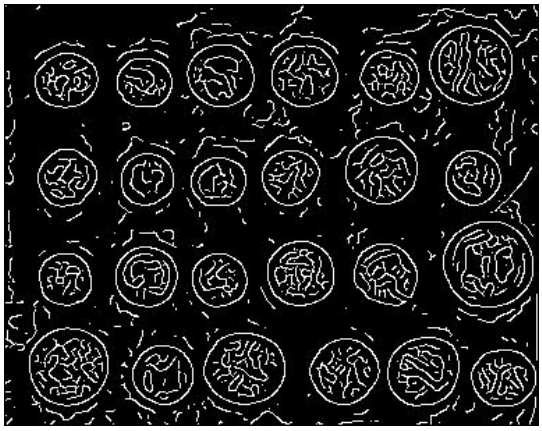
2. V dalším kroku využijeme Soboleova operátoru, který je výhodný z toho důvodu, že nám nejen nalezne velikost gradientu, ale také jeho orientaci hrany.

3. Po aplikování Soboleova operátoru můžou být hrany značně rozmazány, což nám koliduje s třetí podmínkou tohoto algoritmu. Z tohoto důvodu se v tomto kroku snažíme o nalezení lokálních maxim, která nám indikují místa s největší změnou intenzity. Lokální maxima hledáme porovnáním bodů ve směru a proti směru gradientu s vyšetřovaným pixelem. Pokud mají tyto body menší intenzitu, tak se jedná o lokální maximum, tj. pokud bude směr gradientu vertikálně nahoru, porovnáváme pixely pod a nad vyšetřovaným bodem. Pokud se o lokální maximum nejedná, přiřadíme tomuto pixelu hodnotu 0.

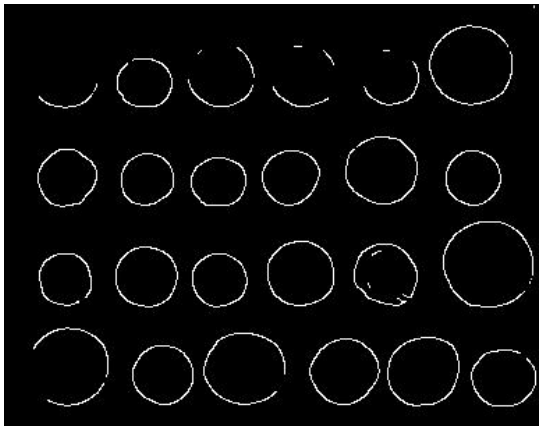
4. V předposledním kroku použijeme prahování postupně pomocí dvou hodnot τ_1 a τ_2, pro které platí, že $\tau_1 < \tau_2$. Tím nám vzniknou dva binární obrazy B_1 a B_2.

5. V posledním kroku se využije tzv. *hystereze*, což znamená, že propojíme segmenty obrazu B_2, tak, aby vytvořily spojitou hranu. Tohoto docílíme tak, že nejprve za body hrany označíme všechny pixely binárního obrazu B_2 a dále za body hrany označíme pixely B_1 takové, které sousedí s pixely $B2$.

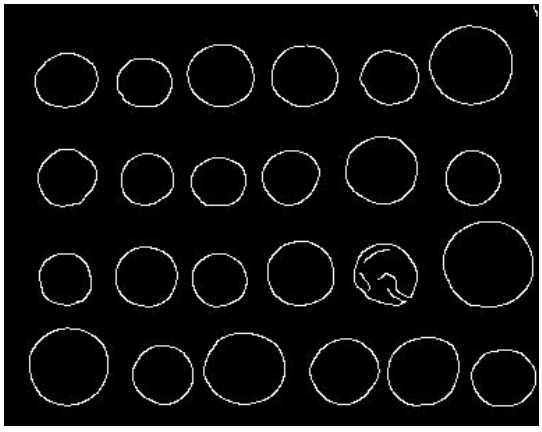
(a) Originální obraz



(b) Prahování s nízkým prahem $\tau_1 = 10$



(c) Prahování s vysokým prahem $\tau_2 = 110$



(d) Výsledný obraz po aplikování hystereze

Obrázek 11: Ukázka použití Cannyho detektoru hran pro $\sigma = 2$

Laplaceův operátor Nyní se podívejme na příklad použití druhé derivace při hledání hran. U této metody se hledá průchod druhé derivace nulou, což je mnohem jednodušší než nalezení extrému. Jedním z operátorů, který druhou derivaci využívá je Laplacián, který má stejné vlastnosti ve všech směrech a tedy je invariantní vůči rotaci. Tento operátor je definován jako

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}.$$

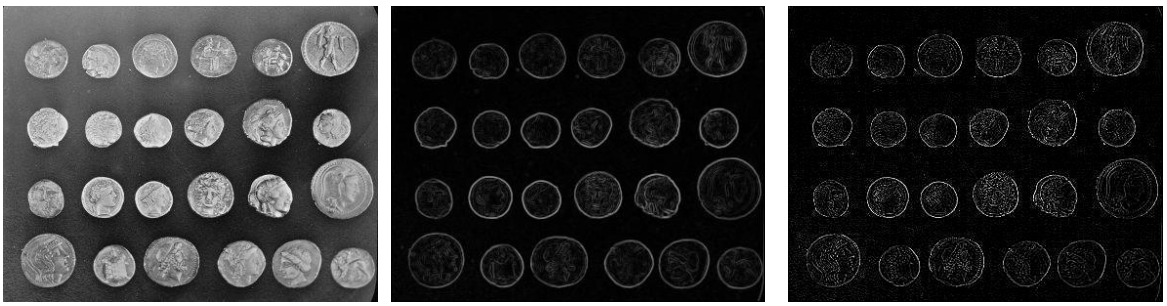
Tato operace může být na obraz opět aplikována pomocí konvolučního jádra $\mathbf{H}_L$, které je v tomto případě, oproti Sobelovu operátoru, pouze jedno. Toto jádro může mít více podob a to například

$$\mathbf{H}_L = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{2.2}$$

v případě impulsové odezvy Laplacianu v případě čtyřsousednosti nebo

$$\mathbf{H}_L = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \tag{2.3}$$

pro osmisousednost. Využití Laplaceovy operace je vhodné zejména v případech, kdy se mění intenzita v obrazu pomalu, jelikož v takové situaci by gradientní operace vytvořila velice širokou hranu. Oproti tomu druhá derivace bude mít průchod nulou v bodě, který je uprostřed této hrany. Na obrázku (12) můžeme vidět porovnání výsledků po použití Sobelova operátoru (12b) a Laplaceova operátoru (12c).



(a) Originální obraz (b) Sobelův operátor (c) Laplaceův operátor

Obrázek 12: Ukázka použití Laplaceova operátoru v porovnání se Sobelovým operátorem

LoG (Laplacian of Gaussian) Jádra Laplaceovy operace jsou, stejně jako u operátorů využívající první derivaci, velmi citlivá na šum, proto je nutné před jejich použitím aplikovat některý z filtrů na redukci šumu, např. Gaussův filtr. Jelikož je konvoluce asociativní operace, můžeme nejprve provést konvoluci vhodného Gaussova filtru g s Laplaceovým operátorem (2.2) nebo (2.3) a poté konvoluci s obrazem. Tento přístup má dvě výhody

- Vzhledem k velikosti Gaussovského a Laplaceova jádra, které jsou většinou mnohem menší než obraz, je tato metoda výpočetně mnohem rychlejší.

- Tímto přístupem můžeme vypočítat tzv. *LoG* (*Laplacian of Gaussian*) *jádro*, čímž docílíme výpočtu pouze jedné konvoluce LoG jádra s obrazem.

Využití LoG operátoru navrhli Marr a Hildreth [10]. Uvažujme obrazovou matici $\mathcal{I} = (\mathcal{P}[i,j])$ pak tvar LoG operátoru můžeme psát s využitím asociativity jako

$$\nabla^2(g * \mathcal{I}) = (\nabla^2 g) * \mathcal{I} = LoG(\mathcal{I}).$$

Odvoďme si nyní analytický tvar tohoto operátoru. Uvažujme Gaussovu funkci ve tvaru (2.1), kde hodnata $A = 1$ a zaveďme substituci $i^2 + j^2 = r^2$, která uvažuje rotační symetrii, pak

$$g'(r) = -\frac{1}{\sigma} r e^{-\frac{r^2}{2\sigma^2}}$$

a

$$g''(r) = \frac{1}{\sigma}\left(\frac{r^2}{\sigma^2} - 1\right) e^{-\frac{r^2}{2\sigma^2}}.$$

Po zpětném dosazení substituce dostaneme LoG operátor ve tvaru

$$\nabla^2 g(i,j) = \left(\frac{i^2 + j^2 - \sigma^2}{\sigma^4}\right) e^{-\frac{i^2+j^2}{2\sigma^2}} \tag{2.4}$$

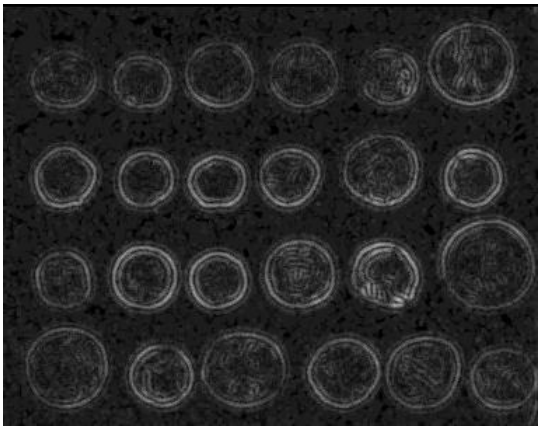
Průchod nulou je v tomto případě dán podmínkou $i^2 + j^2 = 2\sigma^2$. Jádro velikosti 5×5, odpovídající tomuto operátoru pak má podobu

$$\mathbf{H}_{LoG} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}.$$

Nevýhodou tohoto přístupu je přílišné vyhlazování ostrých tvarů, především pak rohů. Ukázku použití *LoG* operátoru můžeme vidět na obrázku (13)

Nutno podotknout, že druhá derivace má průchod nulou i v místech nulové první derivace, ne jen v jejich maximech. To znamená, že se průchody nulou objevují i v místech, kde je obraz homogenní, ne jen v místech hran. Proto je potřeba kromě samotného průchodu nulou testovat, zda-li se v jeho okolí mění znaménko, pokud tomu tak je, bod prohlásíme za hranu.

(a) Originální obraz



(b) Ukázka použití LoG operátoru pro $\sigma = 2$

Obrázek 13: Ukázka použití *LoG* operátoru

2.7 Zpracování binárního obrazu

Jak jsme si uvedli již v kapitole (2.1.1), binárním obrazem rozumíme takový, ve kterém každý pixel nabývá jedné z hodnot 1 nebo 0. Takovéto obrazy jsou zpravidla výsledkem metod segmentace obrazu. V následující části se podíváme na několik operací, které můžeme využít při práci s binárními obrazy před jejich zpracováním. Informace k této kapitole jsme čerpali především z [2][3][9][16].

2.7.1 Matematická morfologie

Uveďme si nejprve dvě základní morfologické operace, které provádějí transformaci jednotlivých objektu v binárním obraze, a to za pomocí tzv. *strukturního elementu*, který je pevně daný.

Definice 2.12 (**Dilatace**)**.** Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i,j])$. Uvažujme dále objekt $O \in \mathcal{I}$ a strukturní element S. Pak *dilatací* $\mathbf{D}$ objektu O strukturním elementem S rozumíme operaci

$$\mathbf{D} = O \oplus S = \{p |\, p = o\ + a, o\ \in O, a\ \in S\}.$$

Definice 2.13 (**Eroze**)**.** Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i,j])$. Uvažujme dále objekt $O \in \mathcal{I}$ a strukturní element S. Pak *erozí* $\mathbf{E}$ objektu O strukturním elementem S rozumíme operaci

$$\mathbf{E} = O \ominus S = \{p |\, a\ \in S : p + a \in O\}.$$

Na obrázku (14) můžeme vidět, jak bude vypadat objekt O po dilataci, resp. erozi. U dilatace tedy dochází k zvětšení počátečního objektu, čehož se využívá například při spojování objektů nebo zaplňování malých děr v objektech. Naopak u eroze dochází ke zmenšení objektu O.

Tyto dvě operace můžeme navzájem i kombinovat. V závislosti na pořadí pak rozlišujeme další dvě morfologické transformace.

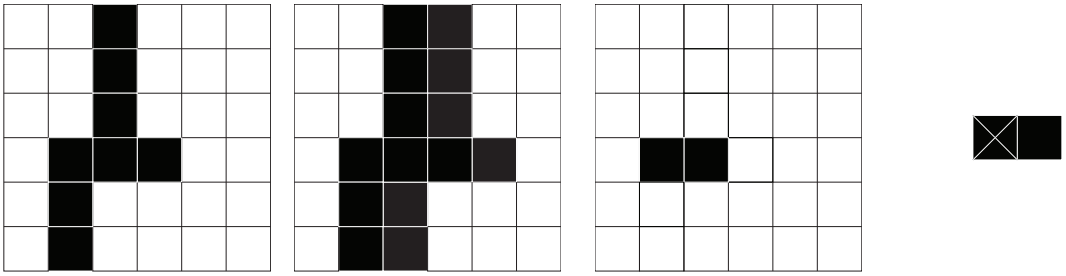
Definice 2.14 (**Otevření**). Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i,j])$. Uvažujme dále objekt $O \in \mathcal{I}$ a strukturní element S. Pak *otevřením* objektu O strukturním elementem S rozumíme operaci

$$O \circ S = (O \ominus S) \oplus S.$$

Definice 2.15 (**Uzavření**). Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i,j])$. Uvažujme dále objekt $O \in \mathcal{I}$ a strukturní element S. Pak *uzavřením* objektu O strukturním elementem S rozumíme operaci

$$O \bullet S = (O \oplus S) \ominus S$$

Význam otevření spočívá v odstranění malých objektů (vzniklých například důsledkem šumu) a detailů bez porušení celkového tvaru objektu O. Aplikací uzavření můžeme docílit propojení objektů, které jsou blízko sebe nebo zaplnění malých děr, opět bez porušení celkového tvaru objektu O.



(a) Původní objekt (b) Objekt po aplikování dilatace (c) Objekt po aplikování eroze (d) Strukturní element

Obrázek 14: Ukázka využití dilatace a eroze za použití strukturního elementu S

2.7.2 Labeling

V mnoha případech nastává situace, kdy se v binárním obraze nalézá více objektů, které se snažíme analyzovat. Abychom byli schopni jednotlivé objekty od sebe při analyzování rozlišit zavedeme *labeling*. Pomocí labelingu jsme schopni jednotlivým objektům obrazu přiřadit specifické označení (běžně se jedná o číslice $1, 2, 3, \dots$) a to na základě zkoumaní propojených pixelů.

Nejběžnějším algoritmem, který se k těmto účelům využívá je tzv. *sekvenční algoritmus*. Tento algoritmus pracuje ve dvou fázích. Uvažujme vstupní obrazovou matici $\mathcal{I}^{in} = (\mathcal{P}[i,j])$ a výstupní obrazovou matici $\mathcal{I}^{new} = (\mathcal{Q}[i,j])$. V první fázi procházíme postupně jednotlivé pixely $\mathcal{P}[i,j]$ obrazové matice $\mathcal{I}^{in}$ a to zleva doprava a z vrchu dolů. Hodnoty jednotlivých pixelů $\mathcal{Q}[i,j]$ obrazové matice $\mathcal{I}^{new}$ vyhodnocujeme na základě následujících podmínek

$$\mathcal{I}^{new} = \begin{cases} 0 & \text{pro } \mathcal{P}[i,j] = 0 \\ l, (l = l + 1) & \text{pro } \mathcal{P}[i-1,j] = 0 \wedge \mathcal{P}[i,j-1] = 0 \\ \mathcal{P}[i-1,j] & \text{pro } \mathcal{P}[i-1,j] \neq 0 \wedge \mathcal{P}[i,j-1] = 0 \\ \mathcal{P}[i,j-1] & \text{pro } \mathcal{P}[i-1,j] = 0 \wedge \mathcal{P}[i,j-1] \neq 0 \\ \mathcal{P}[i,j-1] & \text{pro } \mathcal{P}[i-1,j] \neq 0 \wedge \mathcal{P}[i,j-1] \neq 0 \wedge \mathcal{P}[i-1,j] \neq \mathcal{P}[i,j-1] \\ \mathcal{P}[i,j-1] & \text{pro } \mathcal{P}[i-1,j] \neq 0 \wedge \mathcal{P}[i,j-1] \neq 0 \wedge \mathcal{P}[i-1,j] = \mathcal{P}[i,j-1] \end{cases}.$$

Hodnota l je na počátku rovna 1, kdy tuto hodnotu vždy zvyšujeme na hodnotu $l = l + 1$ při splnění druhé podmínky. Pokud nastane čtvrtá podmínka zapíšeme hodnoty obou pixelů $\mathcal{P}[i-1, j]$ a $\mathcal{P}[i, j-1]$ do takzvané tabulky rovnocennosti. Tato tabulka obsahuje informace o propojených pixelech. Celý tento proces opakujeme pro všechny pixely $\mathcal{P}[i, j]$ obrazové matice $\mathcal{I}$.

V druhé fázi budeme procházet obrazovou matici $\mathcal{I}^{new}$ a to opět po jednotlivých pixelech. Nejprve si však v tabulce rovnocennosti nalezneme nejnižší label l pro každou množinu z této tabulky. Poté každému pixelu obrazové matice $\mathcal{I}^{new}$ přiřadíme hodnotu odpovídající právě tomuto labelu. Tím nám vzniknou jednoznačně rozlišené objekty.

2.8 Momentová metoda

V této části se zaměříme na analýzu jednotlivých objektů, především si určíme jejich základní vlastnosti mezi které patří plocha objektu, těžiště nebo orientace. K těmto účelům nám poslouží momenty jednotlivých objektů, odtud název *momentová metoda*. Zaveďme si nejprve pojem geometrického momentu. Zdrojem těchto informací byly [1][2][22][28].

Definice 2.16 (**Geometrický moment**)**.** Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i, j])$ a dále uvažujme objekt $O \subset \mathcal{I}$. Pak *geometrickým momentem* řádu $p + q$ rozumíme hodnotu

$$m(p, q) = \sum_{\forall \mathcal{P} \in O} i^p j^q \mathcal{P}[i, j]. \tag{2.5}$$

V této části se omezíme výhradně na binárním obrazy, kde nenulové pixely $\mathcal{P}[i, j]$ nabývají pouze hodnoty 1, proto budeme výraz $\mathcal{P}[i, j]$ v rovnici momentu dále vynechávat. Za pomocí geometrického momentu si můžeme zadefinovat první z vlastností objektu O.

Definice 2.17 (**Plocha objektu**)**.** Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i, j])$ a dále uvažujme objekt $O \subset \mathcal{I}$. Pak *plochou objektu* $\mathcal{A}$ rozumíme nultý geometrický moment objektu O. Tedy platí

$$\mathcal{A} = m(0, 0) = \sum_{\forall \mathcal{P} \in O} 1.$$

Díky tomuto vztahu se můžeme plynule přesunout na definici další důležité vlastnosti, kterou budeme potřebovat při dalším zpracování.

Definice 2.18 (**Těžiště**)**.** Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i, j])$ a dále uvažujme objekt $O \subset \mathcal{I}$. Pak bod $T = [\bar{x}, \bar{y}]$, pro který platí

$$\bar{x} = \frac{m(1, 0)}{m(0, 0)}, \qquad \bar{y} = \frac{m(0, 1)}{m(0, 0)},$$

nazveme *geometrickým těžištěm* objektu O.

Těžiště nám bude sloužit k popisu pozice objektu a zároveň si díky němu můžeme zavést tzv. centrální momenty, díky kterým zaručíme invariantnost vůči posunutí.

Definice 2.19 (**Geometrický centrální moment**)**.** Nechť máme obrazovou matici $\mathcal{I} = (\mathcal{P}[i, j])$ a dále uvažujme objekt $O \subset \mathcal{I}$ s geometrickým těžištěm $T = [\bar{x}, \bar{y}]$. Pak *geometrickým centrálním momentem* objektu O řádu $p + q$ rozumíme hodnotu

$$m_c(p, q) = \sum_{\forall \mathcal{P}[i,j]} (i - \bar{x})^p (j - \bar{y})^q.$$

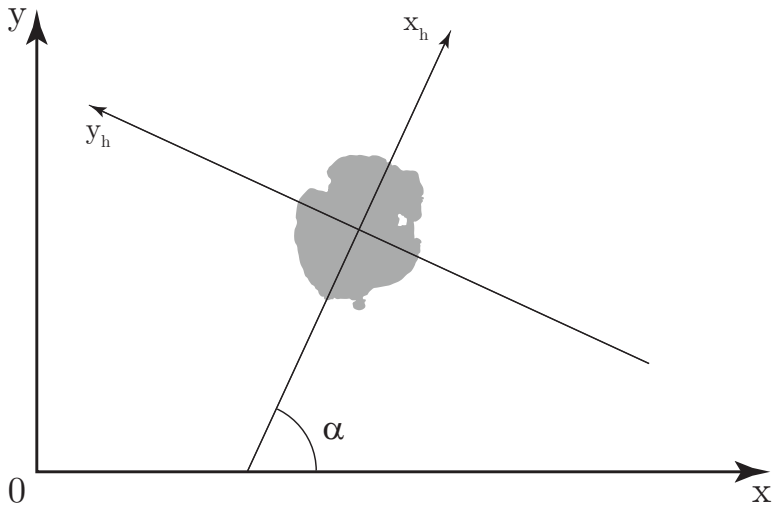
Pro první řády těchto momentů platí základní vlastnost, a to že jsou rovny 0, tedy $m_c(1,0) = m_c(0,1) = 0$. Nyní už nám zbývá zavést pouze poslední typ momentů.

Definice 2.20 (**Centrální geometrický normovaný moment**)**.** *Centrálním geometrický normovaným momentem* řádu $p+q$ rozumíme hodnotu

$$m_{cn}(p,q) = \frac{m_c(p,q)}{[m_c(0,0)]^{\frac{p+q+2}{2}}}.$$

Další z vlastností, kterou si pomocí momentů určíme, je orientace objektu. Abychom si mohli orientaci určit, zaveďme si nejprve tzv. *hlavní soustavu souřadnou*, která splňuje následující podmínky:

1. Plocha objektu $m_h(0,0) = 1$
2. Počátek souřadného systému je v těžišti objektu, tj. $m_h(1,0) = m_h(0,1) = 0$
3. Osy x, y jsou vedeny tak, aby druhý smíšený moment $m_h(1,1) = 0$
4. Pro druhé momenty platí: $m_h(2,0) \geq m_h(0,2)$
5. Třetí moment $m_h(3,0) \geq 0$
6. Souřadný systém je pravotočivý



Obrázek 15: Hlavní souřadný systém a orientace objektu

Zaveďme si nyní tzv. *Legendreovu elipsu.*

Definice 2.21 (**Legendreova elipsa**)**.** Legendreovou elipsou objektu O nazveme elipsu, jejíž osy a osy hlavní soustavy souřadné jsou stejné a pro velikosti její hlavní poloosy A a vedlejší poloosy B platí

$$A = 2\sqrt{m_c(2,0)}, \quad B = 2\sqrt{m_c(0,2)}.$$

Definice 2.22 (**Elongace**)**.** Elongace ε objektu O nám určuje jeho štíhlost a je dána jako

$$\varepsilon = \log_2 \left(\frac{A}{B} \right),$$

kdy hodnota $\varepsilon = 0$ nám udává kruh.

Definice 2.23 (**Orientace objektu**)**.** *Orientací* objektu nazveme úhel α, který svírá osa x_h hlavního souřadného systému objektu s osou x základního souřadného systému objektu.

Znázornění orientace objektu můžeme vidět na obrázku (15). V hlavní souřadné soustavě si zavedeme tzv. *hlavní momenty objektu* řádu $p + q$, značíme $m_h(p, q)$, které nám zaručují invariantnost vůči natočení. Pro tyto momenty platí podle definice (2.5) vztah

$$m_h(p, q) = \sum_{\forall x_h, y_h \in O} x_h^p y_h^q.$$

Nyní si za pomocí známých centrálních normovaných momentů určíme vztah pro orientaci objektu a odvodíme si i některé hlavní momenty. Nejprve si vyjádříme souřadnice hlavní soustavy souřadné pomocí orientace α a souřadnic centrální normované soustvy jako

$$\begin{aligned} x_h &= x_{cn} \cos\alpha + y_{cn} \sin\alpha \\ y_h &= -x_{cn} \sin\alpha + y_{cn} \cos\alpha. \end{aligned} \tag{2.6}$$

Připomeňme si 3. podmínku pro hlavní geometrický souřadný systém, která nám říká, že moment

$$m_h(1, 1) = \sum_{\forall x_h, y_h \in O} x_h y_h = 0 \tag{2.7}$$

Dosaďme nyní vztahy (2.6) do podmínky (2.7) a postupně upravme

$$\begin{aligned} & m_h(1,1) = \sum_{\forall x_h, y_h \in O} x_h y_h = 0 \\ & \sum_{i=1}^{I} \sum_{j=1}^{J} (x_{cn} \cos\alpha + y_{cn} \sin\alpha)(-x_{cn} \sin\alpha + y_{cn} \cos\alpha) = 0 \\ & \sum_{i=1}^{I} \sum_{j=1}^{J} (-x_{cn}^2 \sin\alpha \cos\alpha - x_{cn} y_{cn} \sin^2\alpha + x_{cn} y_{cn} \cos^2\alpha + y_{cn}^2 \sin\alpha \cos\alpha) = 0 \\ & \sin\alpha \cos\alpha \left(\sum_{j=1}^{J} y_{cn}^2 - \sum_{i=1}^{I} x_{cn}^2 \right) + (\cos^2\alpha - \sin^2\alpha) \sum_{i=1}^{I} \sum_{j=1}^{J} x_{cn} y_{cn} = 0 \\ & \sin\alpha \cos\alpha \left[m_{cn}(0,2) - m_{cn}(2,0) \right] + \left(\cos^2\alpha - \sin^2\alpha \right) m_{cn}(1,1) = 0 \end{aligned}$$

a nyní s využitím vztahů pro $\sin 2\alpha$, resp. $\cos 2\alpha$ můžeme poslední vztah upravit na

$$\frac{1}{2} \sin 2\alpha \left[m_{cn}(2,0) - m_{cn}(0,2) \right] = \cos 2\alpha \, m_{cn}(1,1)$$

a tedy po konečné úpravě dostaneme

$$\tan 2\alpha \left[m_{cn}(2,0) - m_{cn}(0,2)\right] = 2m_{cn}(1,1)$$

odkud nám vyplývají dva možné výsledky pro orientaci α v závislosti na centrálních momentech $m_{cn}(2,0)$, $m_{cn}(0,2)$ a to

$$\begin{aligned} \alpha &= \frac{1}{2}\text{arctg}\frac{2m_{cn}(1,1)}{m_{cn}(2,0) - m_{cn}(0,2)}, \quad \text{pro } m_{cn}(2,0) \neq m_{cn}(0,2) \\ \alpha &= \frac{\pi}{4}, \quad \text{pro } m_{cn}(2,0) = m_{cn}(0,2). \end{aligned} \tag{2.8}$$

Odtud nám plyne, že úhel α může nabývat hodnot z intervalu $\left\langle -\frac{\pi}{4}, \frac{\pi}{4} \right\rangle$.

Abychom byli schopni určit hlavní souřadný systém, potřebujeme si odvodit vztahy pro výpočet zbývajících hlavních momentů druhého řádu $m(2,0)$, $m(0,2)$ a moment třetího řádu $m(3,0)$. Podívejme se nejprve na hlavní momenty druhého řádu

$$\begin{aligned} m_h(2,0) &= \sum_{i=1}^{I} x_h^p \\ m_h(2,0) &= \sum_{i=1}^{I}\sum_{j=1}^{J} \left(x_{cn}\cos\alpha + y_{cn}\sin\alpha\right)^2 \\ m_h(2,0) &= \sum_{i=1}^{I}\sum_{j=1}^{J} \left(x_{cn}^2\cos^2\alpha + 2x_{cn}y_{cn}\cos\alpha\sin\alpha + y_{cn}^2\sin^2\alpha\right) \\ m_h(2,0) &= \cos^2\alpha\sum_{i=1}^{I} x_{cn}^2 + 2\cos\alpha\sin\alpha\sum_{i=1}^{I}\sum_{j=1}^{J} x_{cn}y_{cn}) + \sin^2\alpha\sum_{j=1}^{J} y_{cn}^2 \\ m_h(2,0) &= \cos^2\alpha\, m_{cn}(2,0) + 2\cos\alpha\sin\alpha\, m_{cn}(1,1) + \sin^2\alpha\, m_{cn}(0,2). \end{aligned}$$

Podobně si odvodíme i druhý z momentů

$$\begin{aligned} m_h(0,2) &= \sum_{j=1}^{J} y_h^q \\ m_h(0,2) &= \sum_{i=1}^{I}\sum_{j=1}^{J} \left(-x_{cn}\sin\alpha + y_{cn}\cos\alpha\right)^2 \\ m_h(0,2) &= \sum_{i=1}^{I}\sum_{j=1}^{J} \left(x_{cn}^2\sin^2\alpha - 2x_{cn}y_{cn}\cos\alpha\sin\alpha + y_{cn}^2\cos^2\alpha\right) \\ m_h(0,2) &= \sin^2\alpha\sum_{i=1}^{I} x_{cn}^2 - 2\cos\alpha\sin\alpha\sum_{i=1}^{I}\sum_{j=1}^{J} x_{cn}y_{cn} + \cos^2\alpha\sum_{j=1}^{J} y_{cn}^2 \\ m_h(0,2) &= \sin^2\alpha\, m_{cn}(2,0) - 2\cos\alpha\sin\alpha\, m_{cn}(1,1) + \cos^2\alpha\, m_{cn}(0,2). \end{aligned}$$

U hlavního momentu třetího řádu využijeme stejného principu odvození a dostaneme vztah

$$\begin{aligned} m_h(3,0) = &\cos^3\alpha\, m_{cn}(3,0) + 3\cos^2\alpha\sin\alpha\, m_{cn}(2,1) + 3\cos\alpha\sin^2\alpha\, m_{cn}(1,2) + \\ &+ \sin^3\alpha\, m_{cn}(0,3). \end{aligned}$$

Pokud máme tyto hodnoty vypočítány, můžeme hlavní souřadný systém zkonstruovat a to dle následujícího postupu:

1. Určíme hodnoty hlavních momentů $m_h(2,0)$ a $m_h(0,2)$.

2. Jestliže $m_h(2,0) < m_h(0,2)$, zaměníme osy x_h a y_h přičtením hodnoty $\frac{\pi}{2}$ k úhlu α.

3. Určíme hlavní moment $m_h(3,0)$.

4. Jestliže $m_h(3,0) < 0$, musíme otočit kladný směr osy x_h přičtením hodnoty π k úhlu α.

5. Kladný směr osy y_h je dán podmínkou pravotočivého souřadného systému.

2.9 Sekvenční filtrace

Pojmem sekvenční filtrace rozumíme proces, při kterém pracujeme s obrazovou maticí $\mathcal{I}$, která patří do sekvence po sobě jdoucích snímků, a dále s $2n$ obrazovými maticemi z jejího okolí. Tento proces se skládá z celkově pěti kroků, a to:

1. Vytvoření mediánového pozadí $\mathcal{B}_M$ z okolních snímků obrazové matice $\mathcal{I}$.

2. Vytvoření binární masky $\mathcal{M}$ pro kterou platí $|\mathcal{I} - \mathcal{B}_M| > \tau$.

3. Zvětšení binární masky $\mathcal{M}$ pomocí dilatace.

4. Vytvoření průměrového pozadí $\mathcal{B}_P$.

5. Vydělení obrazové matice průměrovým pozadím, tedy $\mathcal{I} \div \mathcal{B}_P$.

Zaveďme si nyní jednotlivé pojmy a vysvětleme si jejich význam. Nejprve se podíváme na oba typy pozadí.

Definice 2.24 (**Mediánové pozadí**)**.** Nechť máme sekvenci po sobě jdoucích snímků. Mějme obrazovou matici $\mathcal{I}^0 = (\mathcal{P}^0[i,j])$ z této sekvence, parametry $n, p \in \mathbb{N}$, kde $n > p$ a $2(n-p)$ obrazových matic z jejího okolí $\mathcal{I}^{-n} = (\mathcal{P}^{-n}[i,j]), \ldots, \mathcal{I}^{-p} = (\mathcal{P}^{-p}[i,j]), \mathcal{I}^{p} = (\mathcal{P}^{p}[i,j]), \ldots, \mathcal{I}^{n} = (\mathcal{P}^{n}[i,j])$. Pak mediánovým pozadím $\mathcal{B}_M = (\mathcal{Q}[i,j])$ obrazové matice $\mathcal{I}^0 = (\mathcal{P}^0[i,j])$ rozumíme

$$\mathcal{Q}[i,j] = median\{\mathcal{P}^m[i,j]\} \quad \forall m,$$

kde $m = -n, \ldots, -p, p, \ldots, n$.

Definice 2.25 (**Průměrové pozadí**)**.** Nechť máme sekvenci po sobě jdoucích snímků. Mějme obrazovou matici $\mathcal{I}^0 = (\mathcal{P}^0[i,j])$ z této sekvence, parametry $n, p \in \mathbb{N}$, kde $n > p$ a $2(n-p)$ obrazových matic z jejího okolí $\mathcal{I}^{-n} = (\mathcal{P}^{-n}[i,j]), \ldots, \mathcal{I}^{-p} = (\mathcal{P}^{-p}[i,j]), \mathcal{I}^{p} = (\mathcal{P}^{p}[i,j]), \ldots, \mathcal{I}^{n} = (\mathcal{P}^{n}[i,j])$. Pak průměrovým pozadím $\mathcal{B}_P = (\mathcal{Q}[i,j])$ obrazové matice $\mathcal{I} = (\mathcal{P}[i,j])$ rozumíme

$$\mathcal{Q}[i,j] = \frac{1}{2(n-p)} \sum_{\forall m} \mathcal{P}^m[i,j],$$

kde $m = -n, \ldots, -p, p, \ldots, n$.

Oba tyto přístupy se snaží o eliminaci pohybujících se objektů, které jsou zachyceny v obrazech naší sekvence snímků. Pojďme se podívat na zbývající tři kroky procesu a vysvětleme si jejich význam. Nejprve si vytvoříme binární masku $\mathcal{M}$ a to pomocí podmínky

$$|\mathcal{I} - \mathcal{B}_M| > \tau,$$

kde operaci odčítání provádíme po jednotlivých pixelech. Nejprve tedy dojde k odečtení mediánového pozadí od obrazové matice $\mathcal{I}$, čímž docílíme zvýraznění všech objektů, které se v obrazové matici nachází, a jsme tedy schopni tyto objekty lépe segmentovat. Poté provedeme segmentaci a to v tomto případě pomocí prahu τ, který můžeme zvolit manuálně nebo můžeme využít některou z automatických metod pro hledání prahu. Tím nám vznikne požadovaná binární maska. Význam vytvořené masky si vysvětlíme později.

V dalším kroku na tuto masku aplikujeme morfologickou operaci dilataci, tak abychom si byli jistí, že bude maskovat všechny části objektů.

Nyní přeskočíme na poslední krok. Před tím, než provedeme příslušnou operaci, upravme si vytvořené průměrové pozadí $\mathcal{B}_P$ a to za využití vytvořené masky. Konkrétně pak proveďme

$$\mathcal{B}_P = \mathcal{B}_M \quad \text{pro } \mathcal{P}[i,j] \in \mathcal{M}.$$

Tím nám vznikne nové pozadí, které si označíme jako $\mathcal{B}_{PM}$. Jedná se tedy o záměnu hodnot pixelů průměrového pozadí, které odpovídají pixelům masky, za hodnoty pixelů mediánového pozadí. Tuto záměnu provádíme, protože mediánové pozadí nám obecně dává lepší výsledky v okolí jednotlivých objektů. Nyní již můžeme přistoupit k poslednímu kroku procesu

$$\mathcal{I} \div \mathcal{B}_{PM}.$$

Tato operace, podobně jako u použitého odčítání, nám zvýrazní všechny objekty, které se v obrazové matici $\mathcal{I}$ vyskytují. Kromě toho se díky tomuto procesu můžeme částečně zbavit nehomogenity v obraze. Stejně jako u odčítání, je i tato operace prováděna po jednotlivých pixelech.

3 Jednoúčelová aplikace

3.1 Vývojové prostředí

Pro vývoj jednoúčelové aplikace byl zvolen programovací jazyk Python, konkrétně pak verze 3.7. Důvodem pro výběr tohoto programovacího jazyka byla jednoduchost syntaxe, kterou zvládne pochopit i méně zkušený uživatel, a možnost objektově orientovaného programování. Důležitou součástí je pak také podpora několika knihoven, které se široce využívají při zpracování obrazu, především pak *scikit-image*, *SciPy*, *OpenCV*, *NumPy* a *matplotlib*.

Zaměřme se nyní na stěžejní knihovnou pro naší práci, kterou je scikit-image, na níž je postavena většina metod a algoritmů použitých v kódu. Scikit-image je knihovna sloužící k zpracovávání obrazu a implementující algoritmy a nástroje využitelné ve výzkumných, vzdělávacích a průmyslových aplikacích. Tato knihovna je vydána pod open source licencí Modified BDS, poskytuje velmi dobře dokumentovanou API v jazyce Python a je neustále vyvíjena aktivním mezinárodním týmem. Tato knihovna je plně dostupná zdarma a bez jakýchkoli omezení. Více informací o tomto projektu je možné nalézt například v [23].

3.2 Příprava dat

Souborem našich dat jsou snímky mikroskopických vláken pohybujících se v modelu lidských plic. Při zpracování dat budeme postupovat ve třech fázích. Nejprve si zvolíme vhodný počáteční obraz, na kterém si připravíme nastavení aplikovaných algoritmů a metod tak, aby vyhovovalo co nejideálněji našim potřebám, což znamená, aby výsledkem byla co nejpřesnější segmentace všech rozeznatelných vláken v pozorovaném obraze. Vhodným počátečním obrazem rozumíme takový, kde se například vlákno či více vláken objeví poprvé, tak abychom jej mohli sledovat po celou dobu jeho pohybu přes zachycenou oblast a najít tak jeho požadované vlastnosti, trajektorii a orientaci.

V druhé fázi budeme postupně procházet náš soubor dat od obrazu, který jsme si zvolili jako počáteční. V každém kroku bude na momentálně pozorovaný obraz aplikováno nastavení algoritmů a metod, tak jak jsme jej zvolili v první fázi. Jakmile se nastavení aplikuje, vypočítají se hledané vlastnosti a ty se pak zapíší do tabulky. Proces se opakuje tak dlouho dokud jej nezastavíme. Výstupem celého procesu je tabulka sledovaných hodnot.

V posledním kroku se zaměříme na zpracování dat, které jsme získali. Nejprve si data vyfiltrujeme tak, aby obsahovala jen pro nás důležité informace. Nakonec data vhodně vizualizujeme.

Celý proces, včetně ukázkových výstupů a popisu jednotlivých použitých metod spolu s jejich nastaveními, si nyní detailněji popíšeme.

3.2.1 Vstupní data

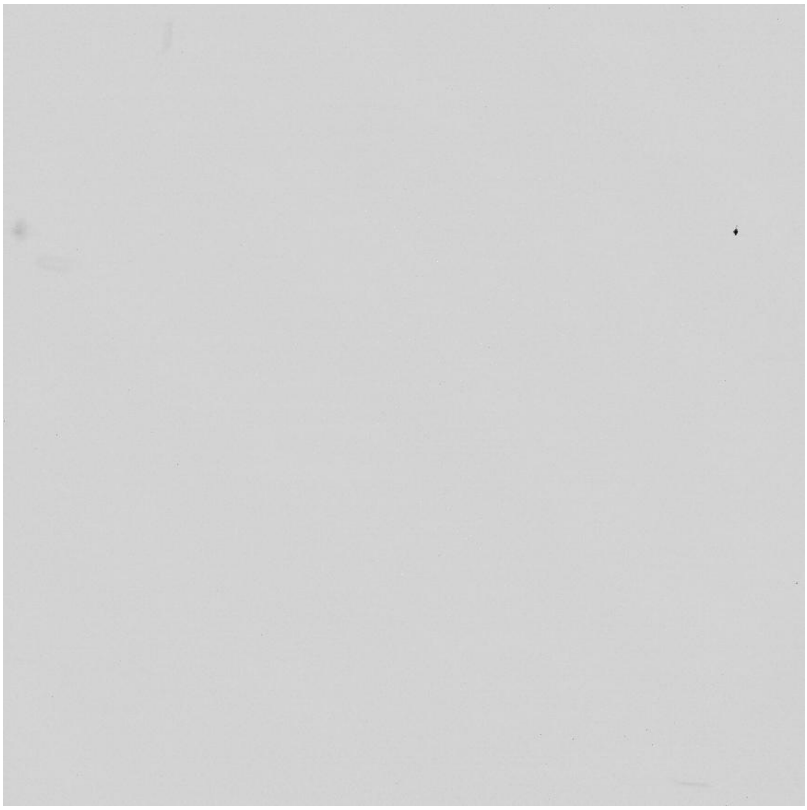
Jak bylo řečeno, v první fázi procesu je potřeba si připravit data ke zpracování, a to nejprve výběrem vhodného výchozího obrazu. Než takový obraz najdeme, podívejme se na naše vstupní data a jejich základní vlastnosti.

Všechna zpracovávaná data byla pořízena vysokorychlostní kamerou, která snímala průlet mikroskopických vláken modelem lidských plic. U primárně využívaného souboru dat byla vzorkováním vytvořena obrazová matice o velikosti 1024×1024, kterou budeme

značit jako $\mathcal{I}^{1024}$. Dále bylo u dat provedeno 12-bitové kvantování, což znamená, že každý pixel $\mathcal{P}[i,j]$ naší obrazové matice $\mathcal{I}^{1024}$ na počátku nabýval hodnot

$$0 \leq \mathcal{P}[i,j] \leq 4095,$$

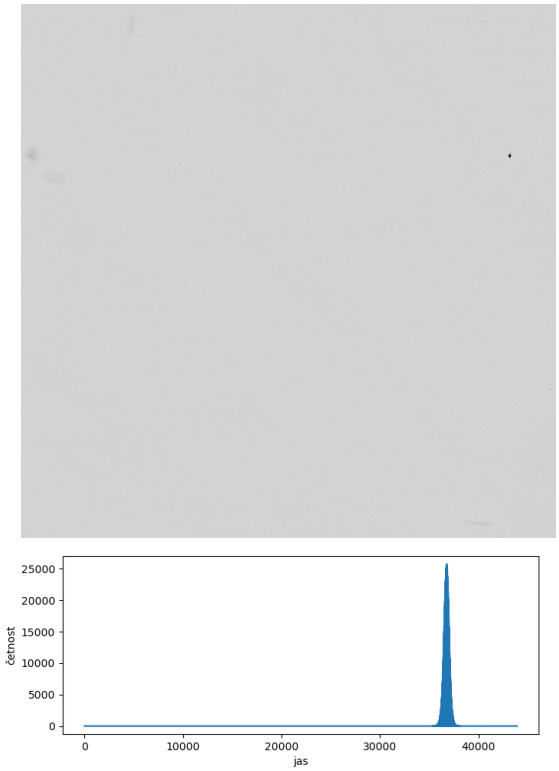
nicméně pro zefektivnění práce, byla všechna data převedena do tzv. datového typu *float* kterým v Pythonu definujeme reálný rozsah hodnot $[0,1]$, resp. $[-1,1]$, ale vzhledem k povaze dat, jsou záporné hodnoty vyloučeny. Ukázku vstupních dat můžeme vidět na obrázku (16).



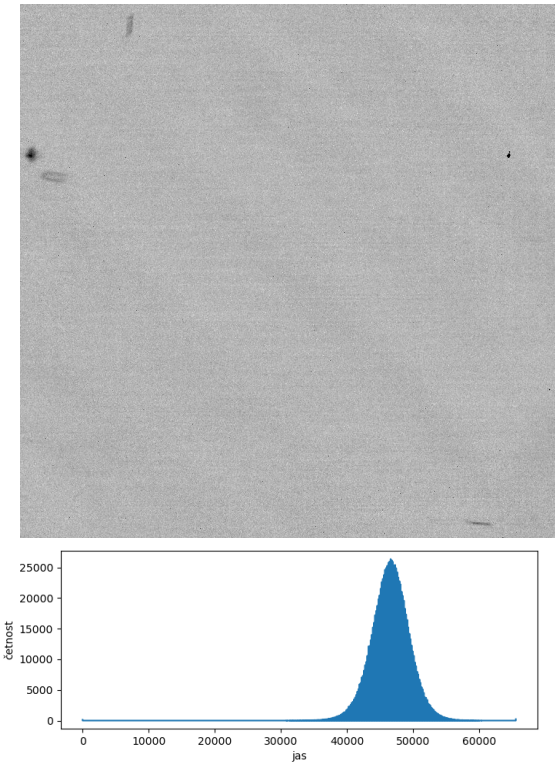
Obrázek 16: Ukázka vstupních dat

3.2.2 Výchozí obraz

Jak je možné vidět na ukázce vstupních dat (16), jednotlivá vlákna jsou z těchto dat jen těžce rozeznatelná, někdy dokonce nejdou na první pohled vůbec vidět, a je tedy velice obtížné takto vybrat vhodný výchozí obraz $\mathcal{I}^{in}$. Pro snadnější rozhodování byla do aplikace implementována metoda ekvalizace histogramu popsaná v kapitole (2.2), díky které můžeme upravit kontrast obrazu do podoby, kdy je možné lépe rozeznat prolétávající vlákna a určit tak ideální výchozí obraz, na který budeme následně aplikovat jednotlivé metody a algoritmy. Nutno podotknout, že ekvalizace histogramu slouží pouze k lepší vizualizaci dat, další zpracování se provádí se základními daty. Ukázku porovnání vstupního obrazu a obrazu s upraveným histogramem můžeme vidět na obrázku (17).



(a) Originální obraz a jeho histogram



(b) Obraz s upraveným histogramem a jeho histogram

Obrázek 17: Porovnání vstupního obrazu a jeho podoby s upraveným histogramem

3.2.3 Sekvenční filtrace

První metodou, kterou použijeme při zpracování, je sekvenční filtrace, kterou jsme popsali v kapitole (2.9). Konkrétně jsme postupovali v následujících krocích:

1. Vytvoření mediánového obrazu s velikostí okolí $2n = 20$ a parametru $p = 3$. Je to z toho důvodu, abychom zaručili, že se nám sledovaná vlákna nebudou na jednotlivých obrazech překrývat. Po vypočítání mediánového pozadí $\mathcal{B}_M = (\mathcal{P}[i,j])$ jsme vytvořili nový obraz $\mathcal{B}_M^{new} = (\mathcal{Q}[i,j])$ jako

$$\mathcal{Q}[i,j] = \text{median } \mathcal{O}_{\square}^{\mathcal{P}}(i,j,5,5).$$

   Tedy aplikovali jsme na něj mediánový filtr a to z toho důvodu, abychom dosáhli lepšího výsledku při vytváření masky.

2. Vytvoření masky $\mathcal{M}$ pomocí odečtení mediánového obrazu od výchozího obrazu a aplikování prahu, který určíme manuálně.

3. Dále využijeme dvou morfologických operací binárního obrazu, které jsou popsány v kapitole (2.7.1), a to otevření a dilataci. V obou případech využijeme strukturního elementu S o velikosti 5×5. A poté vytvořenou masku upravíme následovně

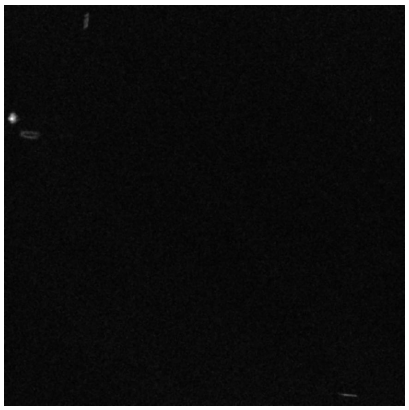
$$\mathcal{M} = (\mathcal{M} \circ S) \oplus S$$

Tedy aplikujeme nejprve otevření, abychom se zbavili menších objektů vzniklých například v důsledku šumu v obraze, a poté využijeme dilatace, tak ať máme jistotu, že jsou všechny objekty O maskou překryty.

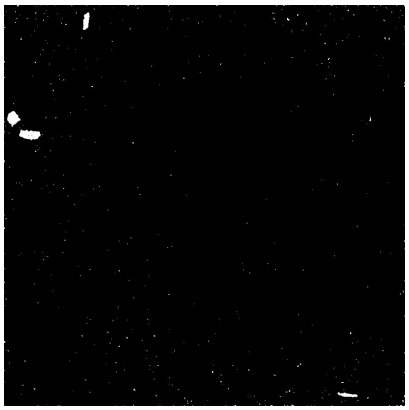
4. Vytvoření průměrového pozadí $\mathcal{B}_P$ s využitím stejných okolních snímků jako jsme použili k tvorbě mediánového obrazu $\mathcal{B}_M$ a záměna příslušných hodnot pixelů s mediánovým pozadím. Tímto vytvoříme v případě našich dat obraz podobný tzv. *flat-fieldu*[2].

5. V posledním kroku provedeme dělení vstupního obrazu $\mathcal{I}^{in}$ upraveným průměrovým pozadím $\mathcal{B}_P$.

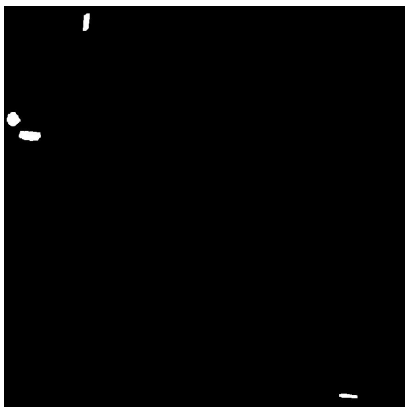
Pro jednoduší práci s obrazovou maticí $\mathcal{I}^{in}$ při vytváření mediánového a průměrového pozadí jsme si obrazovou matici převedli do tvaru řádkové matice o velikosti 1×1048576. Posledním krokem algoritmu docílíme částečného potlačení nehomogenity ve vstupním obraze, která je způsobena poměrně vysokým výskytem šumu. Zároveň se tím, ačkoli to na první pohled nemusí být úplně zřetelné, zvýší kontrast mezi pozadím a sledovanými vlákny. Na obrázku (18) můžeme vidět výstupní obrazy z jednotlivých kroků sekvenční filtrace.



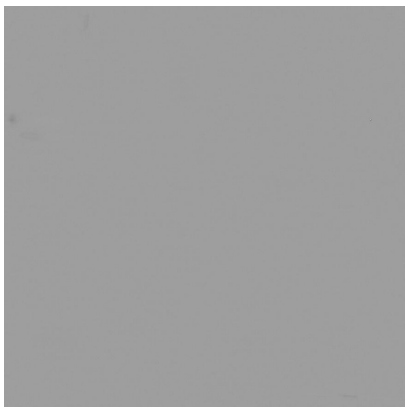
(a) Výsledek odečtení mediánového pozadí od vstupního obrazu



(b) Výsledek manuálního prahování



(c) Maska obrazu po aplikaci otevření a dilatace



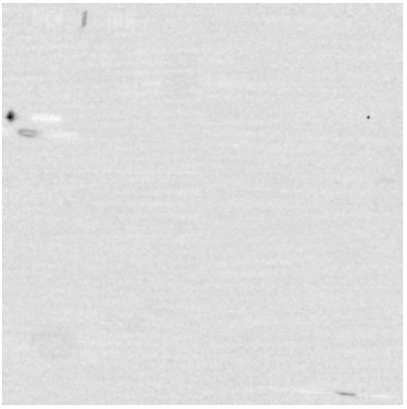
(d) Výsledek dělení originálního obrazu a průměrového pozadí

Obrázek 18: Výsledné obrazy jednotlivých kroků sekvenční filtrace

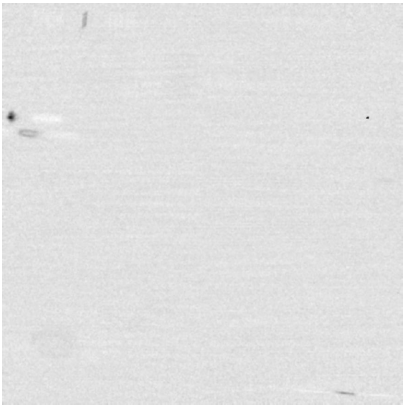
[2]Dokonale homogenní obraz rovnoměrně osvětlené šedé plochy

3.2.4 Filtrace šumu

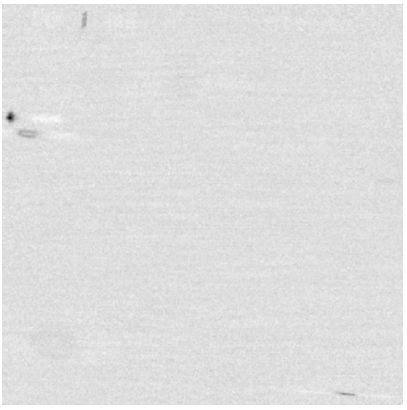
V další fázi se zaměříme na filtraci šumu v obraze za využití některých filtrů popsaných v kapitole (2.4) a (2.5). Do aplikace byly zařazeny čtyři druhy filtrů, dva lineární a dva nelineární, konkrétně se jedná o Gaussův filtr, průměrový filtr, mediánový filtr a bilaterální filtr. Všechny parametry potřebné pro výpočet jednotlivých filtrů jsou volitelné. Jednotlivé výsledky po filtraci, včetně použitých hodnot, můžeme vidět na obrázku (19).



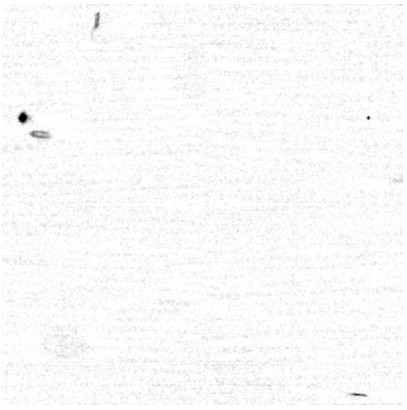
(a) Gaussův filtr s parametrem $\sigma = 2$



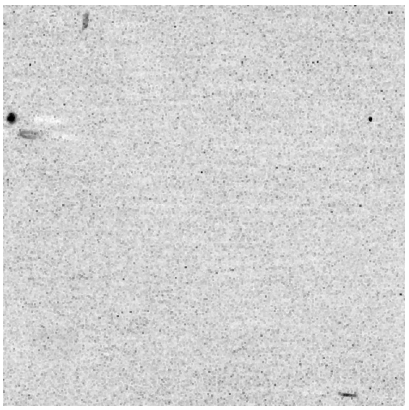
(b) Filtr typu dolní propust s velikostí konvolučního jádra 5×5



(c) Mediánový filtr s velikostí okolí 5×5



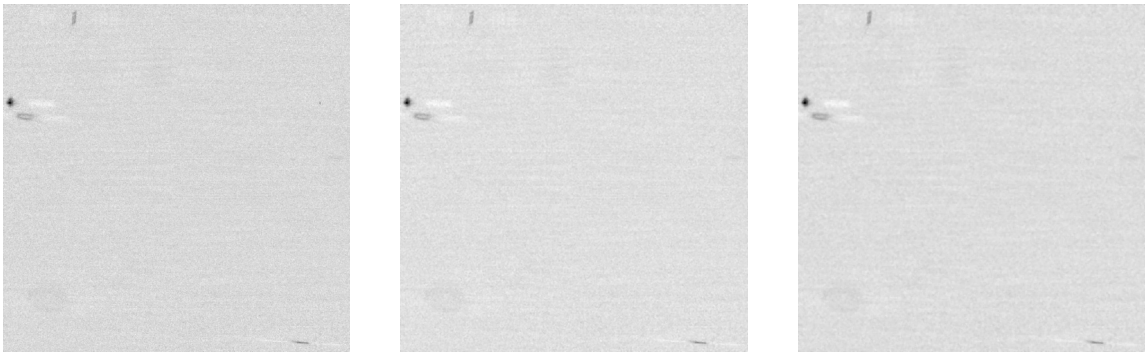
(d) Bilaterální filtr s parametry $\sigma_r = 0.3$ a $\sigma_s = 3$



(e) Filtr typu maxima s velikostí okolí 5×5

Obrázek 19: Porovnání výsledků dosažených aplikací jednotlivých filtračních metod

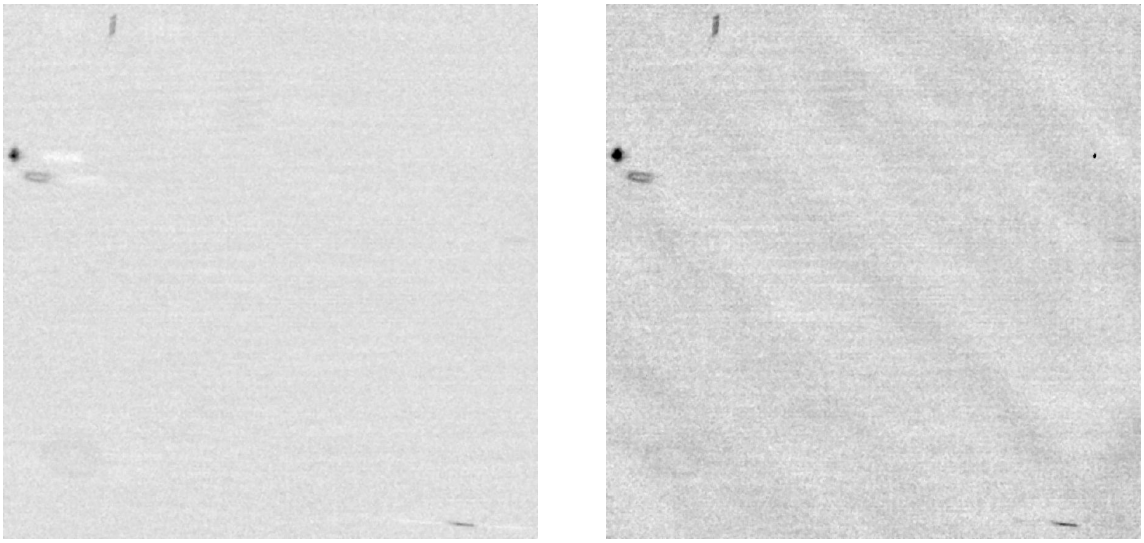
Nejlepších výsledků, a to jak vizuálních tak výpočetních, dosahuje mediánová filtrace, což se dalo očekávat vzhledem k vlastnostem jednotlivých filtrů. Podívejme se proto na výstupy této filtrace při použití různých velikostí okolí. Výsledky můžeme vidět na obrázku (20). Je zřejmé a opět jsme to mohli očekávat, že nejlepšího výsledek, kdy obraz neobsahuje příliš šumu a zároveň nedochází k přílišnému rozmazání, dosáhneme při zvolení velikosti okolí 5×5.



(a) Velikost jádra 3×3 (b) Velikosti jádra 5×5 (c) Velikosti jádra 7×7

Obrázek 20: Porovnání výsledků mediánové filtrace pro různé velikosti jader

V dalším zpracování budeme tedy pracovat právě s tímto obrazem. Pro ukázku můžeme na obrázku (21) vidět situaci, kdybychom aplikovali mediánový filtr přímo na vstupní obraz, aniž bychom použili algoritmus sekvenční filtrace.



(a) Výsledek filtrace na obraz po sekvenční filtraci (b) Výsledek filtrace bez aplikace sekvenční filtrace

Obrázek 21: Srovnání výsledků filtrace mediánovým filtrem s použitím sekvenční filtrace a bez ní

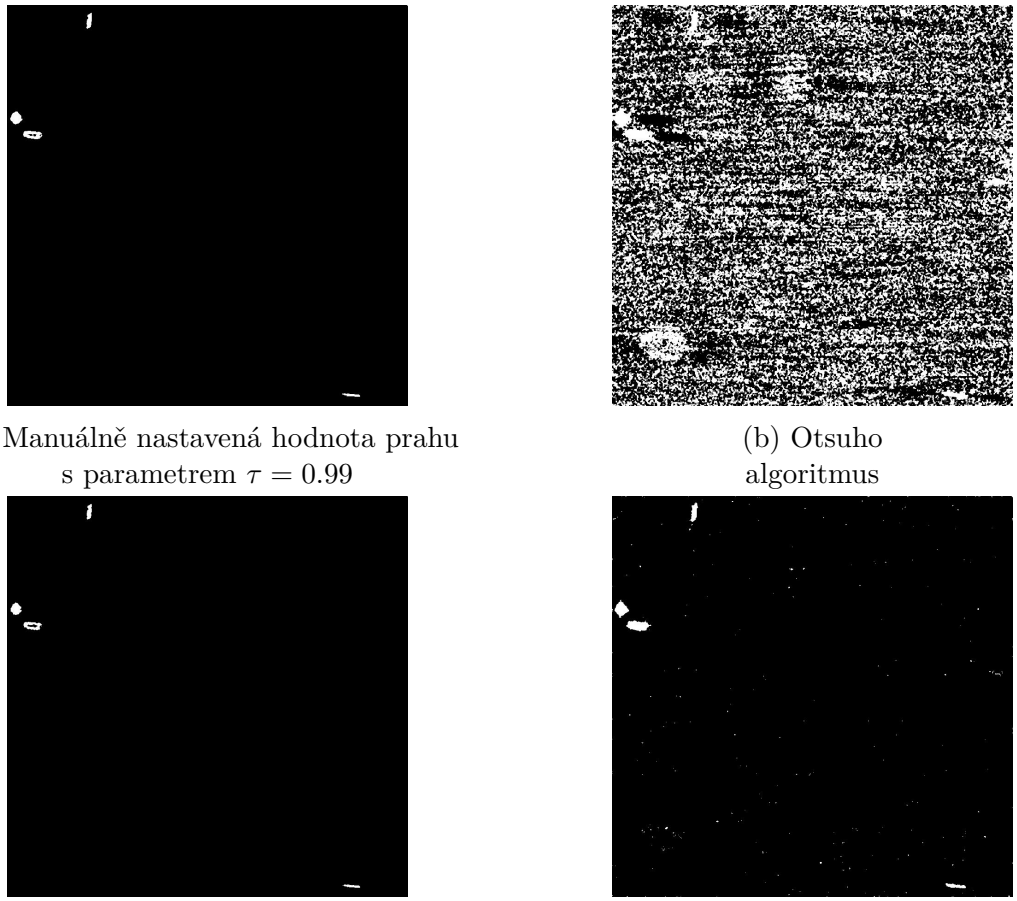
3.2.5 Segmentace obrazu

V předposledním kroku se budeme věnovat stěžejním metodám a to konkrétně zaměřených na segmentaci obrazu. Pro segmentaci využijeme několik algoritmů popsaných v kapitole

(2.6). Jednotlivé metody si detailněji popíšeme a porovnáme získané výsledky.

Manuální prahování Tato metoda je založena na vizuálním hledání ideálního prahu τ obrazu, tak aby nám vznikl binární obraz, ve kterém budou viditelné všechna vlákna nacházející se v obraze.

Automatické metody hledání prahu U tohoto přístupu segmentace využijeme několika automatických algoritmů pro výpočet prahu τ. Konkrétně se jedná o Otsuův algoritmus, metodu ISODATA a trojúhelníkový algoritmus. Více informací o těchto algoritmech je možné najít například v [19][24][25]. Otsův algoritmus jsme popsali mimo jiné také v kapitole (2.6.1). Tyto algoritmy jsou součástí knihovny scikit-image. Srovnání výsledků použitých metod spolu s výsledkem manuálního prahování je možné vidět na obrázku (22). Výsledek po použití Otsuova algoritmu je bohužel nepoužitelný, což je primárně důsledek tvaru histogramu filtrovaného obrazu, který můžeme vidět na obrázku (23), a u kterého si můžeme všimnout, že nemá bimodální rozdělení, které je pro správné fungování této metody podmínkou. Zbývající dvě metody produkují podobné výsledky v porovnání s výsledným obrazem po aplikování manuálního prahu, ačkoli výstup metody ISODATA je přijatelnější.



(a) Manuálně nastavená hodnota prahu s parametrem $\tau = 0.99$

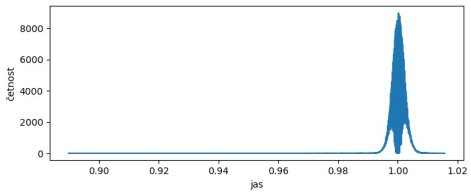
(b) Otsuho algoritmus

(c) Metoda ISODATA

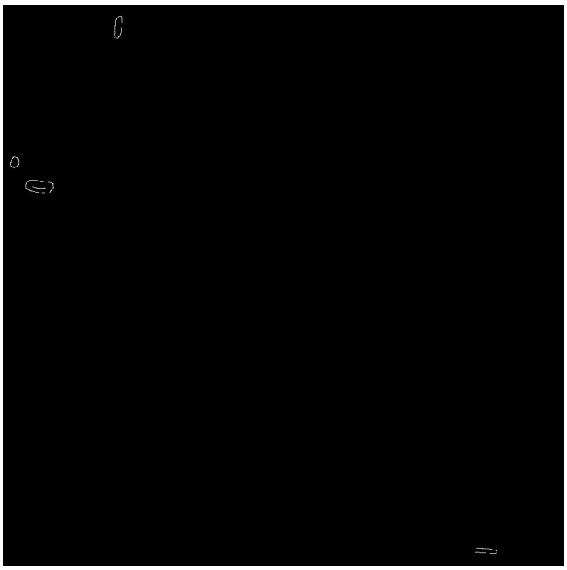
(d) Trojúhelníkový algoritmus

Obrázek 22: Porovnání výsledků automatických algoritmů hledání prahu a obrazu s manuálně nastaveným prahem

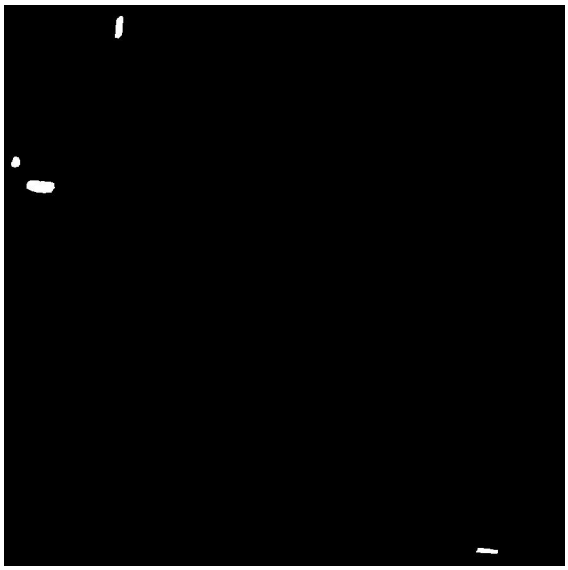
Cannyho hranový detektor Další využitým algoritmem je Cannyho hranový detektor. Jeho postup jsem si popsali v kapitole (2.6.2). U tohoto algoritmu je potřeba zvolit vhodné vstupní σ pro aplikaci Gaussova filtru a dále manuálně najít hodnoty dolního a horního prahu hystereze. Při výpočtu využijeme také masky, kterou jsme si vytvořili při sekvenční filtraci, a kterou omezíme oblast, kde bude algoritmus hledat hrany. Na obrázku (24a) můžeme vidět výsledek po použití tohoto algoritmu. Jde vidět, že některé hrany nejsou dokonale spojité, aplikujeme proto morfologickou operaci uzavření se strukturním elementem ve tvaru čtverce o velikosti 7×7. Tím dosáhneme uzavření hran, v tomto konkrétním případě, dokonce k vytvoření objektů, jak můžeme vidět na obrázku (24b).



Obrázek 23: Histogram obrazu po aplikaci mediánového filtru.



(a) Hrany získané aplikací Cannyho hranového detektoru

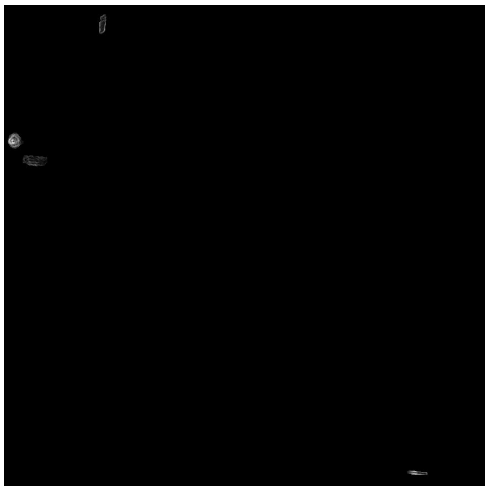


(b) Hrany po aplikaci uzavření

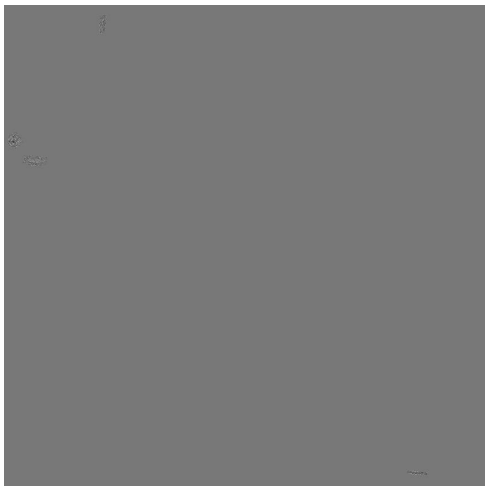
Obrázek 24: Výsledné obrazy po aplikaci Cannyho hranového detektoru a uzavření

Metody založené na hledání hran Posledním přístupem segmentace využitým v aplikaci byly metody, které jsou založené na hledání hran. Konkrétně se pak jednalo o využití hranových operátorů využívajících první a druhou derivaci, které jsme popsali v kapitole (2.6.2). V aplikaci jsme využili Sobelova, Laplaceova, Prewittova a Robertsova operátoru. Na obrázku (25) můžeme vidět výsledek nalezených hran jednotlivými operátory. Při segmentaci byla oblast hledání hran omezena maskou, kterou jsme si vytvořili při sekvenční filtraci. Jak si můžeme všimnout, všechny výsledky, až na Laplaceův operátor, jsou velmi podobné a všechny jsou velmi ovlivněny námi vytvořenou maskou.

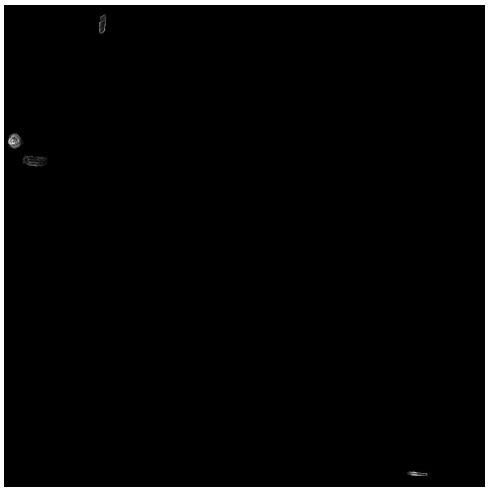
Správnost výsledků segmentace jednotlivých využitých metod vztahujeme k obrazu (22a), který byl vytvořen manuálním nastavením prahu. Nakolik jsou jednotlivé metody využitelné při dalším zpracování si rozebereme v kapitole (3.3).



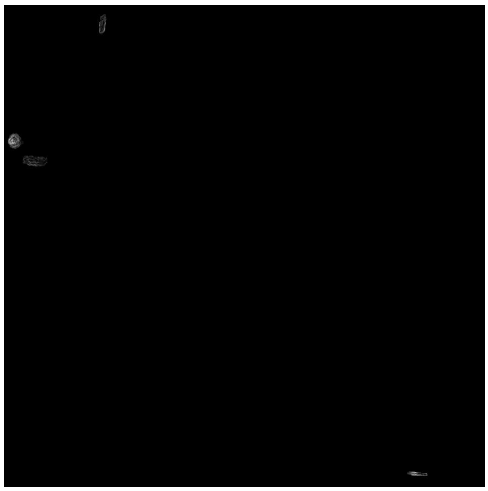
(a) Sobelův operátor



(b) Laplaceův operátor



(c) Prewittův operátor



(d) Robertsův operátor

Obrázek 25: Porovnání výsledků jednotlivých hranových operátorů

3.2.6 Labeling

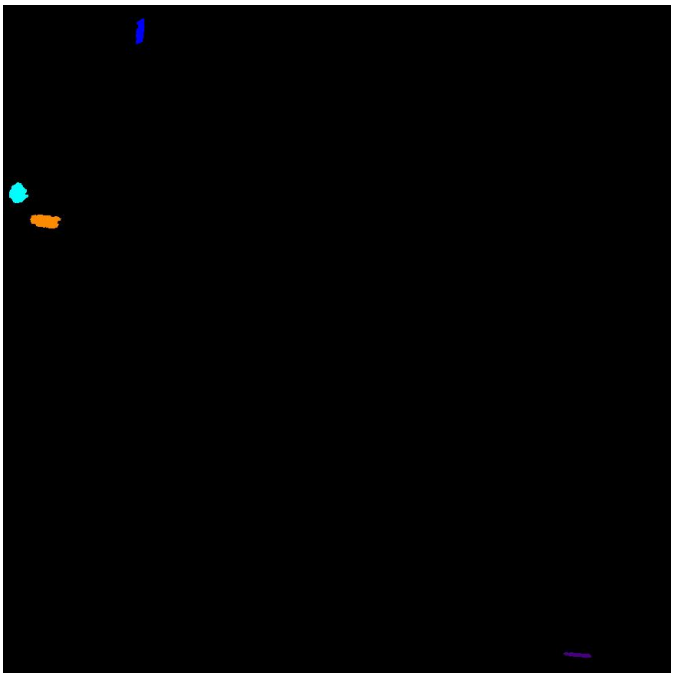
Posledním krokem při přípravě dat je rozlišení jednotlivých vláken, a to za pomocí labelingu, který jsme si uvedli v kapitole (2.7.2). Před tím, než jednotlivým vláknům přiřadíme specifické označení, upravíme si binární obrazy. Nejprve odstraníme objekty O, které nesplňují podmínku

$$\mathcal{A} > \delta,$$

kde $\delta \in \mathbb{N}$ je volitelný parametr s minimální hodnotou 20. Takové objekty O jsou důsledkem šumu a můžeme je vidět například na obrázku (22d). Poté vyplníme díry uvnitř segmentovaných objektů, jelikož jednotlivá vlákna jsou kompaktní.

Na takto upravený binární obraz již můžeme aplikovat metodu labelingu. Získáme tak obrazovou matici $\mathcal{I}$, ve které každému vláknu náleží jiná hodnota a jsou vizuálně odlišeny,

jak můžeme vidět na obrázku (26). Na základě jednoznačného označení každého objektu O jsme pak schopni u jednotlivých vláken vypočítat sledované hodnoty, a to polohu těžiště a orientaci vůči kladnému směru osy x, a to využitím momentové metody, kterou jsme si popsali v kapitole (2.8). Tyto hodnoty jsou poté zapsány do tabulky, jejímž příkladem je tabulka (1). Orientace objektu O je počítána v radiánech.



Obrázek 26: Obraz s již označenými a vizuálně rozlišenými vlákny

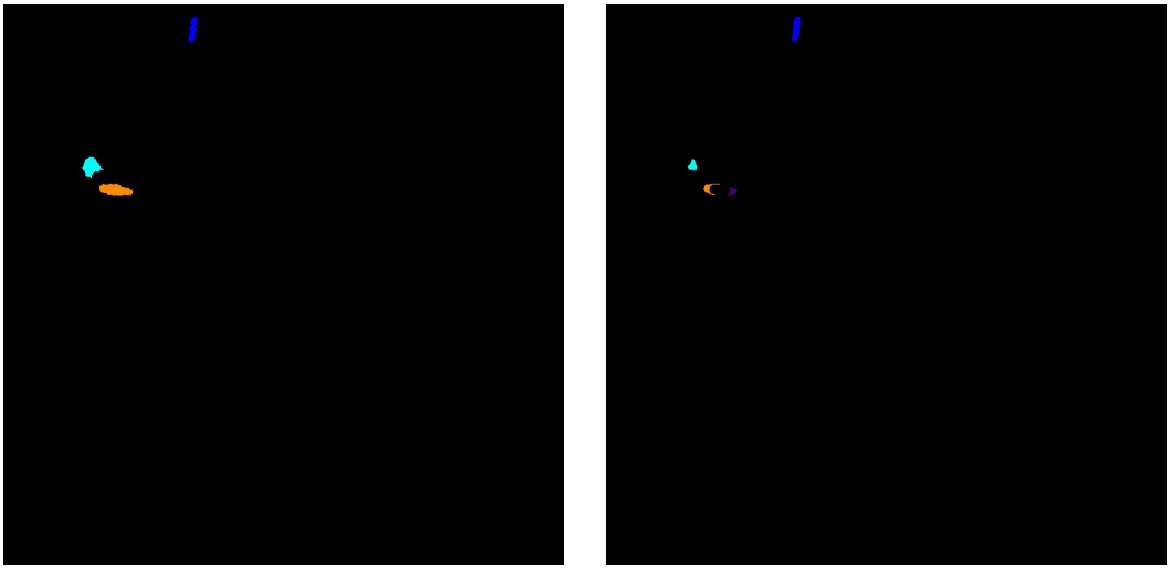
Tabulka 1: Tabulka vypočítaných hodnot pro jednotlivá vlákna

Label	**Color**	**CentroidX**	**CentroidY**	**Area**	**Orientation**	**Elongation**
1	blue	209.84	39.88	377	1.45	1.85
2	cyan	22.57	288.35	637	1.59	0.11
3	darkorange	64.06	330.75	746	3.03	1.33
4	indigo	879.9	995.7	229	3.04	2.79

3.3 Sledování objektů

Při sledování objektů postupujeme postupně naším souborem dat od námi vybraného vstupního obrazu $\mathcal{I}^{in}$. Proces probíhá postupně ve dvou fázích, kdy první fáze je pouze přechod od vstupního obrazu na obraz následující a poté se již iterativně aplikuje fáze druhá. Než přistoupíme k vysvětlení jednotlivých fází, podívejme se nejprve na segmentační metody, které jsme si zmínili v předcházející části a zaměřme se na jejich využitelnost při našem zpracování.

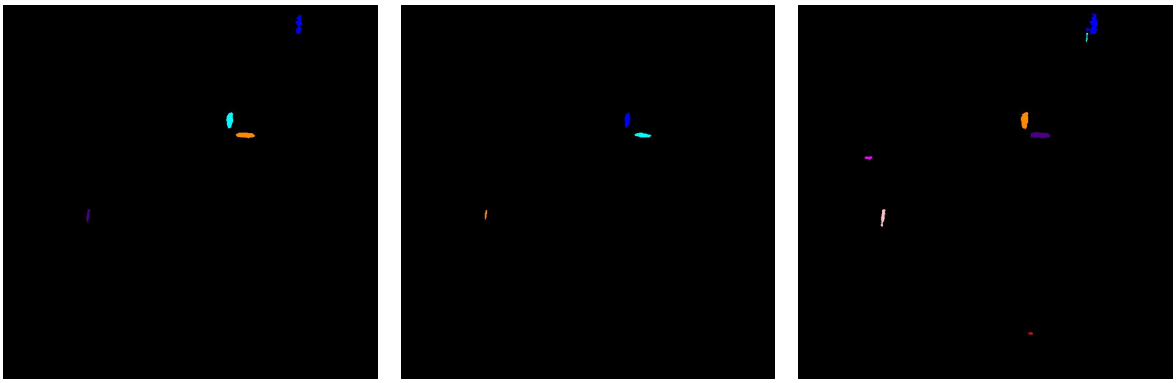
U našich dat narážíme na problém, kdy se pozorovaná vlákna mnohdy ztrácí z hloubky ostrosti. Tento problém se při segmentaci projeví vizuálním rozpadem vlákna, ačkoli ve skutečnosti, jsou vlákna stále vcelku. Největší problém nastává u Cannyho hranového detektoru, který z tohoto důvodu nenalezne všechny hrany vlákna a po jejich vyplnění tak vlákno buď úplně zmizí nebo z něj zbyde jen část, jak můžeme vidět na obrázku (27).



(a) Manuální prahování (b) Cannyho hranový detektor

Obrázek 27: Porovnání obrazů segmentovaných manuálním prahováním a Cannyho hranového detektoru

U metod založených na hledání hran se projevila jejich vysoká citlivost na šum a ačkoli byly segmentované oblasti omezeny pouze na oblast masky vytvořené v sekvenční filtraci, nebyly výsledky postačující. U automatických metod hledáni prahu se ukázaly jako využitelné dvě metody a to metoda ISODATA a trojúhelníkový algoritmus. Bohužel, opět, ani u těchto metod nebyly výsledky konstantní skrze celý soubor dat v porovnání s použitím manuálního prahování. Porovnání těchto výsledků můžeme vidět na obrázku (28). Z těchto důvodů jsme se nakonec uchýlili k metodě manuálního prahu, u které budeme v případě nutnosti iterativně měnit hodnotu prahu tak, abychom dosáhli co nejlepších výsledků.



(a) Manuální prahování (b) Metoda ISODATA (c) Trojúhelníkový algoritmus

Obrázek 28: Porovnání výsledků segmentace metodou ISODATA, trojúhelníkovým algoritmem a manuálním prahováním

Přesuňme se nyní již k samotnému sledování objektů. Jak bylo řečeno, proces se skládá ze dvou fází. V první fázi si nejprve aplikujeme úpravy, které jsme provedli u vstupního obrazu $\mathcal{I}^{in}$, na následující obrazovou matici v našem souboru dat, což tedy znamená

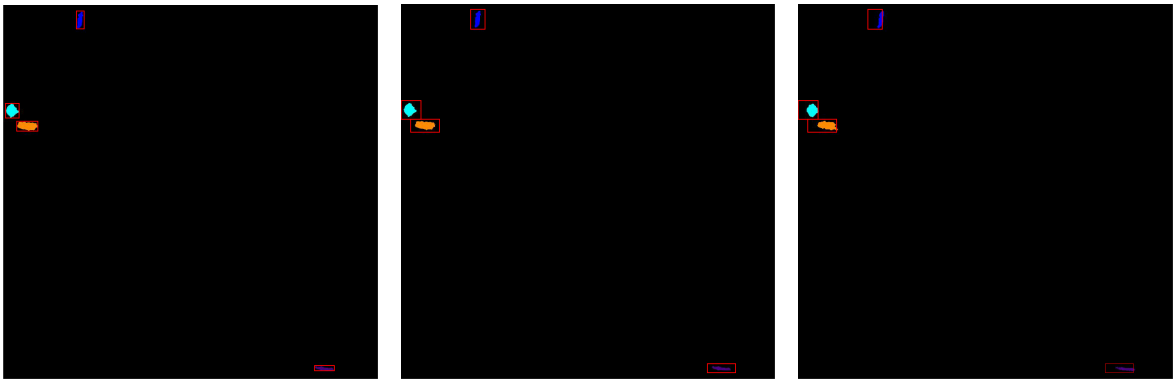
konkrétně na obrazovou matici $\mathcal{I}^{in+1}$. Uvažujme, že obě obrazové matice obsahují stejný počet vláken z. Ve vstupním obraze $\mathcal{I}^{in}$ nalezneme nejmenší obdélníkové okolí $\mathcal{O}_k^{in}$, kde $k = 1, 2, \ldots, z$, jednotlivých vláken $O_k^{in} \subset \mathcal{I}^{in}$, takové že

$$\mathcal{P}[i,j] \in \mathcal{O}_k^{in}, \quad \forall \mathcal{P}[i,j] \in O_k^{in}.$$

Tyto okolí nyní zvětšíme v kladném i záporném směru obou os a to postupně o m a n pixelů, kde $m, n \in \mathbb{N}_0$. Vznikne nám tak rozšířené okolí, které si označíme jako $\mathcal{O}_k^{new}$. Nyní se podíváme na obrazovou matici $\mathcal{I}^{in+1}$, ve které budeme zkoumat oblasti, odpovídající okolím $\mathcal{O}_k^{new}$ ze vstupního obrazu $\mathcal{I}^{in}$. Tyto oblasti musí obsahovat právě jedno vlákno, resp. část jednoho vlákna. Pokud tomu tak není, je potřeba vhodně upravit hodnoty m, n. Pokud v oblasti vlákno nalezneme, prohlásíme jej za vlákno původní. U objektů $\mathcal{O}_k^{in+1} \in \mathcal{I}^{in+1}$ pak spočítáme hledané parametry, za předpokladu, že je splněna podmínka

$$\mathcal{A}_k > \delta,$$

kde $\delta \in \mathbb{N}$ je opět volitelný parametr větší než 20. Uvedený proces je nezbytný pro správné fungování druhé fáze. Ukázku první části procesu můžeme pak vidět na obrázku (29), včetně znázorněných okolí.



(a) Nejmenší okolí jednotlivých vláken

(b) Zvětšené okolí jednotlivých vláken

(c) Prohledávaná oblast v obrazové matici $\mathcal{I}^{in+1}$

Obrázek 29: Výsledné obrazy po aplikaci Cannyho hranového detektoru a uzavření

Druhá část procesu začíná podobně jako ta první, a to přípravou nového obrazu, v tomto případě $\mathcal{I}^{in+l}$, pro jednoduchost budeme v dalším textu značit jako $\mathcal{I}^l$, kde $l = 2, 3, 4, \ldots$. Obecně pak nejprve opět nalezneme v obrazové matici $\mathcal{I}^{l-1}$ nejmenší obdélníkové okolí $\mathcal{O}_k^{l-1}$ všech vláken $O_k^{l-1} \subset \mathcal{I}^{l-1}$, pro které platí, že

$$\mathcal{P}[i,j] \in \mathcal{O}_k^{l-1}, \quad \forall \mathcal{P}[i,j] \in O_k^{l-1}.$$

Uvažujme těžiště $T_k^{in+l-2} = [\bar{x}, \bar{y}]$ vláken $O_k^{l-2} \subset \mathcal{I}^{l-2}$ a těžiště $T_k^{in+l-1} = [\bar{x}, \bar{y}]$ odpovídajících vláken $O_k^{l-1} \subset \mathcal{I}^{l-1}$, pak

$$[m_k, n_k] = T_k^{in+l-1} - T_k^{in+l-2}, \quad k = 1, 2, \ldots, z$$

kde z nám určuje počet vláken. Nalezená okolí $\mathcal{O}_k^{l-1}$ budeme v tomto případě posouvat v příslušných směrech, a to o vzdálenosti m_k, n_k. Posunuté okolí si označíme jako $\mathcal{O}_k^{new+l-1}$. V obrazové matici $\mathcal{I}^l$ nyní prohledáváme oblast, která odpovídá posunutému

okolí $\mathcal{O}_k^{new+l-1}$. V tomto okolí očekáváme výskyt vlákna, pokud tomu tak je, a je toto vlákno v tomto okolí právě jedno, můžeme vypočítat potřebné vlastnosti a zapsat je do tabulky. Nicméně může nastat situace, kdy se v této oblasti objeví více objektů, což je právě způsobeno vylétnutím vlákna z hloubky ostrosti. V takovémto případě přistoupíme k iterativnímu zvyšování počátečního prahu τ_{in} a to o danou hodnotu

$$\tau_{new} = \tau_{in} + 0,005.$$

Tato iterace se ukončí ve dvou případech:

1. V prohledávaném okolí nalezneme pouze jedno vlákno.
2. Hodnota $\tau_{new} > 1$.

V prvním případě si můžeme přímo vypočítat potřebné informace nalezeného vlákna, u druhé možnosti nejprve prohlásíme všechny nalezené objekty za jediný a poté přistoupíme k výpočtu. Takto nalezené hodnoty budou touto skutečností zkresleny. Po výpočtu jsou samozřejmě opět všechny informace uloženy do tabulky tak, ať s nimi můžeme dále pracovat. Po analyzování všech vláken změníme hodnotu prahu na původní τ_{in} a v případě nutnosti si ji opět iterativně upravíme, a to z toho důvodu abychom nepracovali se zbytečně vysokou hodnotou prahu. Celý proces druhé fáze opakujeme tak dlouho, než přijde signál z uživatelského prostředí aplikace o zastavení sledování.

Zaměřme se ještě na výpočet orientace α, která je podle odvozených výsledků z kapitoly (2.8) pouze v rozsahu $\left\langle -\frac{\pi}{4}, \frac{\pi}{4} \right\rangle$. V mnoha programovacích jazycích nicméně nalezneme funkci

$$\text{atan2}\left(\frac{y}{x}\right) = \text{arctg}\left(\frac{y}{x}\right).$$

Obecně nám tato funkce vrací hodnoty z intervalu $(-\pi, \pi\rangle$, ale v našem konkrétním případně, vzhledem ke vztahu (2.8), to bude interval $\left\langle -\frac{\pi}{2}, \frac{\pi}{2} \right\rangle$, a to vzhledem k 0-té ose, která odpovídá ose y kartézského souřadného systému, v proti směru hodinových ručiček. Abychom získali úhel, vzhledem k ose x kartézského souřadného systému, stačí k vypočtené hodnotě přičíst $\frac{\pi}{2}$, tím určíme orientaci vůči kladnému směru osy x, a to v intervalu $(0, \pi\rangle$. Vlákna se nicméně mohou během pohybu otáčet v rozsahu $\langle 0, 2\pi\rangle$. Z tohoto důvodu jsme zavedli funkci, která nám zkoumá průchod přes osu x, a to obecně za podmínky

$$\left|\alpha_k^l - \alpha_k^{l-1}\right| > \frac{\pi}{2}, \tag{3.1}$$

pro $k = 1, 2, \ldots, z$ a $l = 1, 2, \ldots$, kde z nám označuje počet objektů v obraze a α_k^l, resp. α_k^{l-1} jsou postupně orientace příslušící objektu $O_k^l \subset \mathcal{I}^l$, resp. $O_k^{l-1} \subset \mathcal{I}^{l-1}$. Hodnota na pravé straně nerovnice (3.1) byla určena z pozorování a říká nám, že nepředpokládáme otočení vlákna mezi dvěma snímky o více než $\frac{\pi}{2}$. Pokud taková situace nastane, a tedy podmínka (3.1) bude splněna, pak orientaci α^l vlákna k spočítáme jako

$$\alpha_k^l = \alpha_k^l + \pi.$$

Tuto hodnotu k orientaci připočítáváme do té doby, než opět dojde k situaci, kdy je splněna podmínka (3.1).

4 Výsledky

Jak bylo řečeno v minulé kapitole, všechny vypočtené informace o jednotlivých vláknech ze všech obrazových matic, které při sledování objektů prošly pozorovanou oblastí, byly postupně zapisovány do tabulky, jejímž příkladem je tabulka (1). Aplikace umožňuje částečnou práci s daty, uloženými v této tabulce, konkrétně pak je možné data v tabulce upravovat, mazat nevhodné řádky, provést automatickou filtraci, export tabulky do formátu pro další zpracování a v neposlední řadě je možné vykreslit trajektorii a průběh změny orientace jednotlivých vláken. Podívejme se postupně na jednotlivé možnosti a uveďme si jejich význam.

4.1 Automatická filtrace dat

Automatická filtrace dat probíhá ve dvou krocích. V první fázi nalezne všechna vlákna s určitým označením, která nesplňují podmínku

$$n_k > 20, \quad \text{pro } k = 1, 2, \ldots, z$$

kde n_k nám udává počet záznamů v tabulce o vlákně k a z je celkový počet vláken, která se během sledování v obrazových maticích objevila. U těchto vláken je pak nabídnuta možnost na smazání veškerých dat o tomto konkrétním vlákně. Pokud se rozhodneme data vymazat, již nebudou při případném vykreslení výsledků vizualizovány.

Druhá část filtrace je zaměřena na pozice těžiště $T_k = [\bar{x}_k, \bar{y}_k]$ jednotlivých vláken. Pro každé vlákno porovnáváme nalezené souřadnice těžiště s jejich předpokládanou polohou. Konkrétně pak iterativně procházíme nalezené hodnoty a přičítáme k nim průměrný přírůstek, který budeme pro x-ovou složku značit h_k^{normX}. Nejprve si vždy spočítáme vzdálenost h_k^m, pro kterou ve směru x-ové souřadnice platí

$$h_k^m = h_k^{m-1} + \bar{x}_k^l - \bar{x}_k^{l+1}$$

kde $m = 1, 2, \ldots$ a $l = 0, 1, \ldots$, kdy $\bar{x}_k^0$ rozumíme pozice těžišť jednotlivých vláken, ve vstupním obraze $\mathcal{I}^{in}$, který byl zaveden v kapitole (3.2.2) a $h_k^0 = 0$. Průměrný přírůstek h_k^{normX} pak dostaneme jako

$$h_k^{normX} = \frac{h_k^m}{m},$$

Přírůstek je poté přičten k hodnotě $\bar{x}_k^{l+1}$ a zkoumáme podmínku

$$\left| \bar{x}_k^{l+1} + h_k^{normX} - \bar{x}_k^{l+2} \right| < 0,2 h_k^{normX}. \tag{4.1}$$

Obdobně si zavedeme podmínku i pro y-ovou souřadnici, u které budeme průměrný přírůstek značit h_k^{normY}, jako

$$\left| \bar{x}_k^{l+1} + h_k^{normY} - \bar{x}_k^{l+2} \right| < 0,5 h_k^{normY}. \tag{4.2}$$

Pokud není podmínka splněna, jsou příslušné buňky tabulky označeny, abychom s nimi případně mohli dále pracovat. Hodnoty $0,2$ a $0,5$ na pravých stranách nerovností (4.1) a (4.2) byly určeny na základě měření. Důvodem, proč některá data nevyhovují nerovnostem (4.1) nebo (4.2) může být více, ale jedná se především o problém nastíněný v kapitole

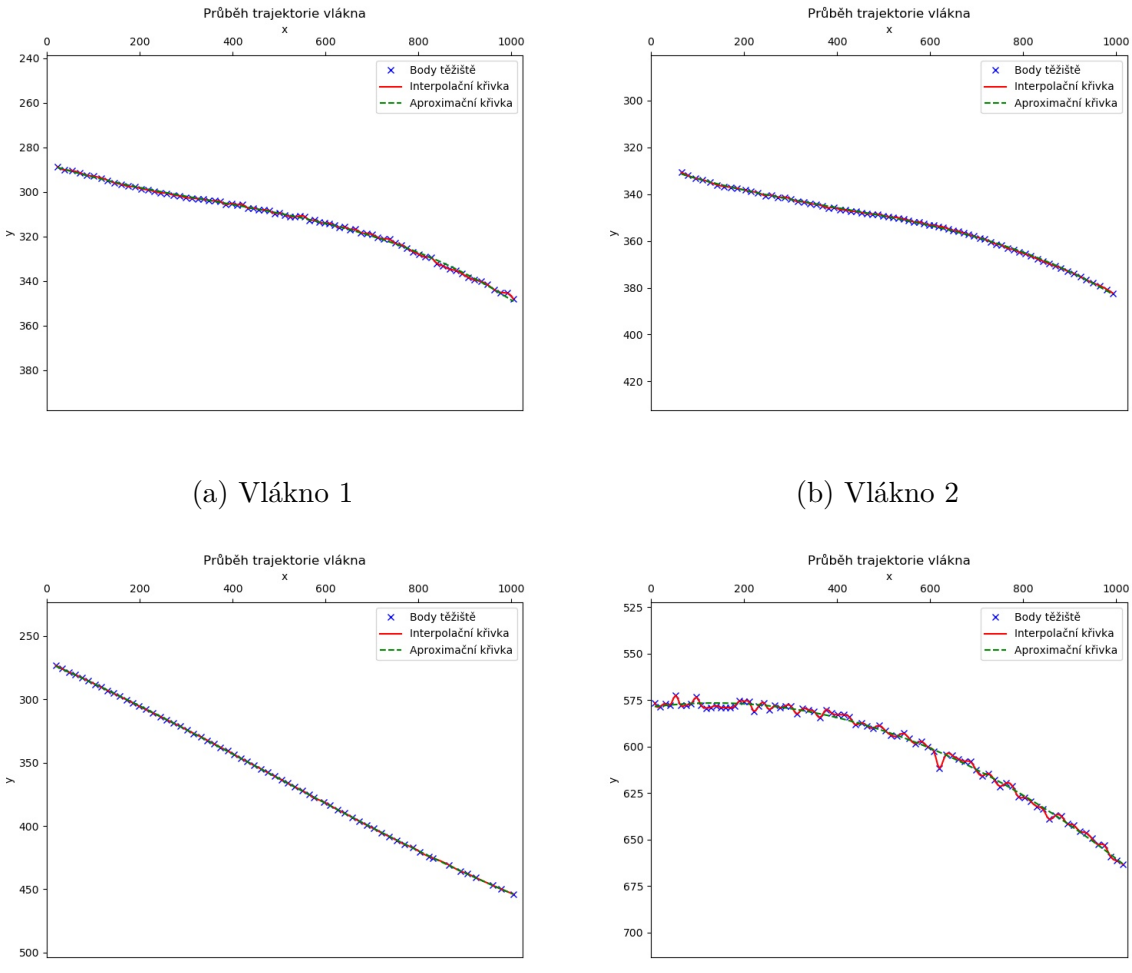
(3.3), a to vylétnutí vlákna, či vláken z hloubky ostrosti kamery, kdy se vlákna můžou rozdělit na více částí nebo například změnit svou plochu, čímž se silně ovlivní poloha těžiště. Všechna vlákna, u kterých nalezneme souřadnice těžiště, které neodpovídají našim podmínkám máme opět možnost odstranit z tabulky dat, ty pak dále nebudou v případě vizualizace vykresleny.

4.2 Export dat

Druhou možností, kterou nám aplikace nabízí, je export dat pro jejich další zpracování v aplikacích třetích stran, konkrétně se jedná o export ve formátu *.xlsx*, který můžeme dále zpracovávat například v softwaru *Microsoft Excel*. Jak bylo řečeno, data jsou exportovány ve tvaru, který vidíme v tabulce, tedy včetně veškerých úprav, které jsme s daty provedli.

4.3 Vizualizace výsledků

Věnujme se již nyní nejpodstatnější části a to vizualizaci dat, která se provádí přímo v samotné aplikaci.



(a) Vlákno 1

(b) Vlákno 2

(c) Vlákno 3

(d) Vlákno 4

Obrázek 30: Ukázka trajektorie jednotlivých vláken

Trajektorie Podívejme se nejprve na trajektorie sledovaných vláken. Příklady čtyř trajektorií vláken ve sledované oblasti můžeme vidět na obrázku (30).

Grafy znázorňují části obrazové matice, ve kterých jsou znázorněny jednotlivé naměřené souřadnice těžišť $T_k = [\bar{x}_k, \bar{y}_k]$, interpolační křivku, k jejímuž vytvoření jsme využili již zmíněný kubický splajn a aproximační křivku, kterou jsme vytvořili pomocí metody nejmenších čtverců a polynomu čtvrtého stupně. Pohyb všech vláken byl zleva doprava, můžeme tedy vidět, že všechny trajektorie mají „klesající" charakter, nicméně ve spojitosti s obrazovou maticí se samozřejmě nelze bavit vyloženě o klesání. Tato skutečnost je primárně dána nastavením kamerového systému, jelikož vlákna proudí skrze model průdušky pouze jedním směrem. Podívejme se nyní na jednotlivé grafy.

Nejméně zajímavý je pro nás graf na obrázku (30d), který má na první pohled téměř lineární průběh. U grafů na obrázku (30a) a (30b) si můžeme povšimnou mírného zvlnění. Spolu s tím, u těchto vláken pozorujeme pomalejší klesání, tento fakt si vysvětlujeme větší velikostí těchto vláken v porovnání s vláknem z grafu (30d).

U poslední grafu, z obrázku (30c), se může na první pohled zdát že docházelo k fluktuaci vlákna a tedy i jeho těžiště z důvodu skokovitých změn těchto bodů. Toto je částečně pravda, jelikož k fluktuaci docházelo, nicméně byla způsobena problémem, který jsme nastínili v sekci (3.3). Konkrétně se nám ztratila část vlákna, z důvodu nízkého prahu a v závislosti s tím se posunula i poloha těžiště. Tento problém můžeme sledovat po celé délce křivky.

Jak si můžeme všimnout ve většině případů, nám interpolační křivka splývá s aproximační, nicméně v případech, kdy se nám vlákno částečně ztrácí, získáme pomocí aproximační křivky lepší výsledky.

Orientace Ukažme si nyní i čtyři příklady, na kterých můžeme vidět průběh orientace a elongace jednotlivých vláken. Tyto orientace, až na jeden příklad, budou odpovídat vláknům, jejichž trajektorie jsme si ukázali na obrázku (30). Příklady průběhu orientace a elongace můžeme vidět na obrázku (31).

Grafy nám znázorňují průběh změny orientace a elongace jednotlivých vláken v závislosti na jejich poloze v obrazové matici.

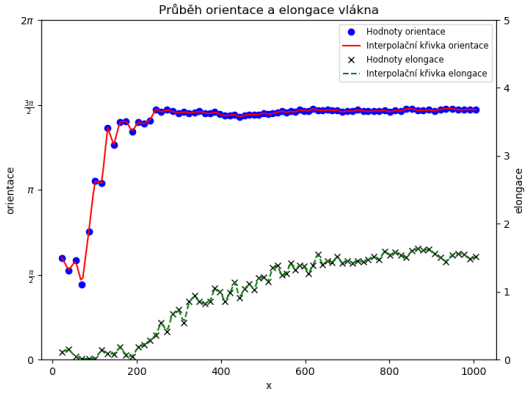
Začneme u grafu na obrázku (31b), který odpovídá trajektorii z obrázku (30b), kde můžeme pozorovat téměř konstantní orientaci okolo hodnoty π, tedy u tohoto vlákna nepozorujeme žádné otočení, což nám dokazuje i průběh elongace, která je téměř konstantní.

Zajímavější průběh nicméně pozorujeme na obrázku (31a), jenž se pojí s trajektorií (30a), a dochází v něm ke skokové změně orientace, což může být způsobeno buď otočením vlákna a nebo výskytem většího množství šumu v okolí vlákna, který může při segmentaci tvar vlákna ovlivnit. Více informací o tom, čím byla skoková změna způsobena, nám může přinést průběh elongace, jejíž hodnota se v místech skokové změny orientace pohybuje okolo 0, což nám značí, že vlákno mělo téměř kruhový tvar, tedy s velkou pravděpodobností zde došlo k otočení vlákna. Konkrétně se jednalo o otočení z polohy $\frac{\pi}{2}$ až na hodnotu $\frac{3\pi}{2}$, na které toto vlákno již zůstalo.

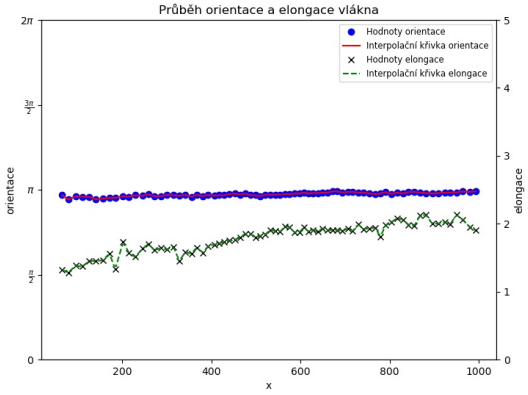
Opačný případ můžeme pozorovat na obrázku (31c), kde je hodnota orientace konstantní a až na úplném konci dochází k výrazným skokovým změnám. První ze znázorněných skoků byl způsoben skutečností, že se nám vlákno z jednoho snímku vytratilo, resp. neprošlo segmentací a objevilo se opět na dalším, kdy se u něj znovu vypočetla hodnota orientace, bez závislosti na předchozích hodnotách. Tohoto si můžeme povšimnout i na obrázku trajektorie (30c), ve kterém můžeme vidět, že zde opravdu v pravé části chybí

některé body těžiště. Následující skoky orientace pak byly způsobeny pouze přechodem přes x osu. Skutečnost, že se vlákno opravdu během pohybu neotočilo si můžeme ověřit i grafem elongace, která nebyla nižší než 1.

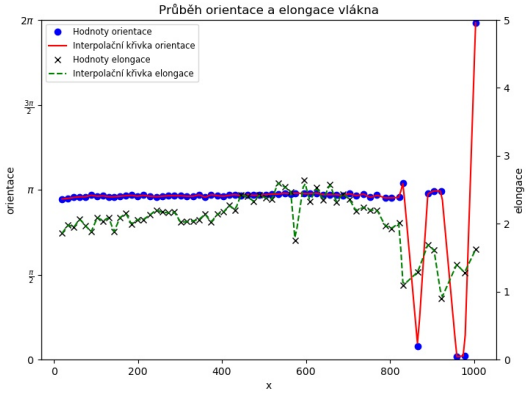
Poslední ukázka z obrázku (31d) není spojená s žádnou z trajektorií. Na tomto grafu můžeme vidět ukázku vlákna, které taktéž změní svou orientaci. Konkrétně se zde jednalo o změnu orientace z vertikální polohy do polohy horizontální. O otočení opět vypovídá i graf elongace, která se v místech skokových změn blížila 0.



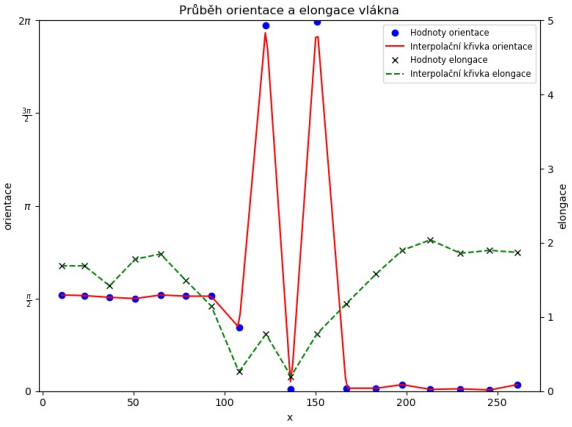
(a) Vlákno 1



(b) Vlákno 2



(c) Vlákno 3



(d) Vlákno 5

Obrázek 31: Ukázka průběhu orientace a elongace jednotlivých vláken

Závěr

Tato diplomová práce se zabývala sledováním objektů v obrazech pořízených vysokorychlostní kamerou a jejím hlavním cílem bylo vytvoření jednoúčelové aplikace, která bude tyto obrazy zpracovávat. Objekty, které jsme analyzovali, byla mikroskopická skleněné vlákna, která se pohybovala v replice lidské průdušnice. Tím bylo simulováno vdechování těchto vláken při manipulaci se skelnou vatou Supafil® Loft, ze které byla vlákna vytvořena, a která se používá k izolaci. Vlákna můžou mít při dlouhodobějším vdechování toxikologické následky, a proto je velmi důležité zabývat se jejich pohybem. Do budoucna je u vláken podobného typu potenciál farmaceutický, kdy tato vlákna mohou být potenciálně nosiči farmaceutických látek a při znalosti jejich pohybu můžou být tyto látky dopravovány na specifická místa. Samotné výsledky dosažené v této diplomové práci budou sloužit k dalšímu zpracování a analýze.

V první kapitole této práce jsme se věnovali potřebnému matematickému základu pro zpracování obrazu a dalšího využití.

Druhá kapitola se nejprve věnovala zavedení obrazové matice. Poté jsme se zaměřili na zavedení lineárních a nelineárních filtrů spolu s jejich příklady a využitím. Dále jsme si ukázali několik metod segmentace obrazu a také operace spojené s binárním obrazem. V neposlední řadě jsme si zavedli tzv. sekvenční filtraci a momentovou metodu.

Předposlední kapitola se věnovala námi vytvořené aplikaci, která byla vytvořena v programovacím jazyce Python a je objektově orientována. Kapitola se především věnuje popisu procesu zpracování obrazů a porovnání výsledků, kterých jsme díky aplikaci dosáhli použitím konkrétních metod. Aplikace má vlastní uživatelské prostředí a využívá různých metod pro zpracování obrazové informace, které jsme si uvedli v textu. Uživatel si pro zpracování může vybrat z několika různých přístupů a nebo využít doporučené metody zpracování, které jsme uvedli na konci této kapitoly.

V poslední kapitole jsme si ukázali několik možností, jak lze pomocí naší aplikace pracovat se výstupními daty, ale především jsme si graficky znázornili zkoumané parametry několika sledovaných objektů, a to jejich trajektorii a orientaci, spolu s elongací. U výsledků jsme okomentovali jejich průběh a uvedli jsme možné příčiny rozdílných vývojů. Vizualizované výsledky byly docíleny pomocí doporučených metod zpracování z předcházející kapitoly. U ostatních metod jsme dostávali velice zkreslené výstupy z důvodu horší použitelnosti na našich konkrétních datech, které obsahovaly velké množství šumu a to i po jeho filtraci.

Ve zpracovávaných datech můžeme pozorovat velice velký obsah šumu, což je způsobeno primárně rychlostí závěrky kamery, a tedy nedostatkem světla při sledování vláken. Tento šum nám velmi komplikuje následné pozorování a zpracování, nicméně i přes tento fakt, se nám podařilo kombinací určitých metod zpracování obrazu získat, pomocí námi vytvořené aplikace, výsledky vhodné k další analýze.

Seznam použité literatury

Reference

[1] Pratt, William K.: *Digital Image Processing (Fourth Edition) PIKS Scientific Inside* [online]. 4th ed. New Jersey: Wiley-Interscience, 2007, [cit. 2020-04-03]. Dostuné z: https://www.academia.edu/30586970/Digital_Image_Processing_4th_Edition_-_William_K_Pratt

[2] Jain, R., Kasturi, R., Scunck, Brian G.: *Machine Vision* [online]. McGraw-Hill, Inc., ISBN: 0-07-032018-7, 1995, [cit. 2020-04-03]. Dostupné z: https://www.academia.edu/download/30794723/MachineVision.pdf

[3] Sojka, E., Gaura, J., Krumnikl, M.: *Matematické základy digitálního zpracování obrazu* [online]. Ostrava: Vysoká škola báňská–Technická univerzita, 2011, [cit. 2020-05-21]. Dostupné z: http://www.kiv.zcu.cz/ novyp/zvi/digitalni_zpracovani_obrazu.pdf

[4] Lizal, F., Maly, M., Farkas, A., Pech, O., Misik, O., Jicha, M., Cabalka, M., Sujanska, L. E., Štarha, P., Kedajova, K., Jadelsky, J.: *Visualization of the flow of mocrometre-sized glass fibres in a replica of the human trachea.* Experimental Fluid Mechanics, Franzensbad, 2019, 275 - 278.

[5] Bolvik, A.: *The Essentianl Guide to Image Processing* [online]. Elsevier Inc., ISBN: 978-0-12-374457-9, 2009, [cit. 2020-04-04]. Dostupné z: https://www.academia.edu/36070136/The_essential_guide_to_image_processing

[6] Bezvada, V.,: *Dvojrozměrná diskrétní Fourierova transformace a její použití* I. Státní pedagogické nakladatelství, Praha, 1988.

[7] Fiřt, J., Holota, R.: *Digitalizace a zpracování obrazu* [online]. 2020, [cit. 2020-04-07]. Dostupné z: https://www.researchgate.net/publication/267235327_Digitalizace_a_zpracovani_obrazu

[8] Fajmon, B.; Růžičková, I.: Matematika 3 [online]. Skriptum FEKT VUT v Brně, 2005, [cit. 2020-06-11]. Dostupné z: http://www.umat.feec.vutbr.cz/ novakm/matematika3.pdf

[9] Petrou, M.M., Petrou, C.: *Image processing: the Fundamentals* [online]. John Wiley & Sons, 2010, [cit. 2020-04-07]. Dostupné z: https://www.academia.edu/10176202/Image_Processing_The_Fundamentals_Image_Processing_The_Fundamentals_Second_Edition

[10] Marr, D., Hildreth, E.: *Theory of Rdge Detection* [online]. Proceedings of the Royal Society of London. Series B. Biological Sciences 207, 1980, 187-217, [cit. 2020-04-06]. Dostupné z: https://royalsocietypublishing.org/doi/pdf/10.1098/rspb.1980.0020

[11] Hambal, A. M., Dr. Pei, Z., Ishabailu, F. L.: *Image Noise Reduction and Filtering Techniques* [online]. International Journal od Science and Research, ISNN: 2319-7064, 2015, 226-230, [cit. 2020-04-04]. Dostupné z: https://www.ijsr.net/archive/v6i3/25031706.pdf

[12] Patidar, P., Gupta, M., Srivastava, S., Nagawat, A. K.: *Image De-noising by Various Filters for Different Noise* [online]. International Journal of Computer Applications, Volume 9 - No.4, November 2010, 45-50, [cit. 2020-04-04]. Dostupné z: https://pdfs.semanticscholar.org/9ed6/7bdec2ead156435989912d36f23a777925ec.pdf

[13] Hu, Q., He, X., Zhou, J.: *Multi-Scale Edge Detection with Bilateral Filtering in Spiral Architecture* [online]. Pan-Sydney Area Workshop on Visual Information Processing. ACM Digital Library, 2004, [cit. 2020-04-03]. Dostupné z: https://opus.lib.uts.edu.au/bitstream/10453/1754/3/2004001944.pdf

[14] Oh, B. M., Chen, M., Dorsey, J., Durand, F.: *Image-based modeling and photo editing* [online]. Proceedings of the 28th annual conference on Computer graphics and interactive techniques, 2001, 433-442, [cit. 2020-06-11]. Dostupné z: https://dl.acm.org/doi/pdf/10.1145/1281500.1281604

[15] Tomasi, C., Manduchi, R.: *Bilateral filtering for gray and color images* [online]. Sixth international conference on computer vision (IEEE Cat. No. 98CH36271), 1998, 839-846, [cit. 2020-06-11] Dostupné z: https://ieeexplore.ieee.org/abstract/document/710815/.

[16] ŠRÁMEK, J., et al.: *Získávání a analýza obrazové informace* [online]. Masarykova univerzita v Brně, Lékařská fakulta–Biofyzikální ústav, 2011, [cit. 2020-06-08]. Dostupné z: https://www.med.muni.cz/biofyz/Image/ucebnice.pdf

[17] Kaur, D.: *Noise Types and Various Removal Techniques* [online]. International Journal of Advanced Research in Electronics and Communication Engineering, Volume 4, Issue 2, February 2015, 226-230, [cit. 2020-04-03]. Dostupné z: http://ijarece.org/wp-content/uploads/2015/02/IJARECE-VOL-4-ISSUE-2-226-230.pdf

[18] Kaur, D., Kaur, Y.: *Various Image Segmentation Techniques: A Reviw* [online]. International Journal of Computer Science and Mobile Computing, Vol. 3, Issue. 5, May 2014, 809-814, [cit. 2020-04-03]. Dostupné z: https://ijcsmc.com/docs/papers/May2014/V3I5201499a84.pdf

[19] Liu,D., Yu, J.: *Otsu method and K-means* [online]. 2009 Ninth International conference on Hybrid Intelligent System, Beijing, China, 2009, 344-349, [cit. 2020-04-03]. Dostupné z: ieeexplore.ieee.org/abstract/document/5254345

[20] Yuheng, S., Hao, Y.: *Image Segmentation Algorithms Overview* [online]. 1. SiChuan University, SiChuan, ChengDu, 2017, [cit. 2020-04-03]. Dostupné z: https://arxiv.org/abs/1707.02051

[21] Dr. Vijayarani, S., Vinupriya, M.: *Performance Analysis of Canny and Sobel Edge Detection Algorithms in Image Mining* [online]. International Journal of Innovative Research in Computer and Communication Engineering, vol. 1, issue 8, October 2013, 1760-1767, [cit. 2020-04-03]. Dostupné z: http://www.academia.edu/download/51942832/29.IJIRCCE-Vinupriya-Performance_Analysis_of_Canny_and_Sobel_Edge_Detection_.pdf

[22] Štarha, P.: *Aplikace numerických metod zpracování obrazové informace.* Habilitační práce oboru „Aplikovaná matematika“, Brno, VUT-FSI, Ústav matematiky, 2015, 105s, [cit. 2020-04-08].

[23] van der Walt S., Schönberger J.L., Nunez-Iglesias J., Boulogne F., Warner J.D., Yager N., Gouillart E., Yu T.: *scikit-image: image processing in Python*[online]. 2014 [cit. 2020-05-22].

[24] Ridler, T.W., Calvard, S.: *Picture Thresholding Using an Iterative Selection Method* [online]. IEEE Transactions on Systems, Man, and Cybernetics, vol. 8, no. 8, Aug. 1978, 630-632, [cit. 2020-06-06]. Dostupné z: https://ieeexplore.ieee.org/document/4310039

[25] Zack, G. W., Rogers, W. E., Latt, S. A.: *Automatic measurement of sister chromatid exchange frequency.* [online]. Journal of Histochemistry and Cytochemistry, 25(7), 1977, 741–753, [cit. 2020-06-06]. Dostupné z: https://journals.sagepub.com/doi/pdf/10.1177/25.7.70454

[26] KLÍČ, A., VOLKA, K., DUBCOVÁ, M.: *Fourierova transformace s příklady z infračervené spektroskopie* [online]. [cit. 2020-06-17]. Dostupné z: http://old.vscht.cz/mat/Pavel.Pokorny/students/ft/skripta/Four.pdf

[27] Čermák, L., Hlavička, R.: *Numerické metody.* Akademické nakladatelství CERM, sro, 2008.

[28] Rudolfová, Z.: *Vytváření nanostruktur na površích látek hybridními metodami.* [online]. Doktorská práce, Ústav fyzikálního inženýrství, Brno, VUT FSI, 2018, [cit. 2020-06-17]. Dostupné z: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=176983

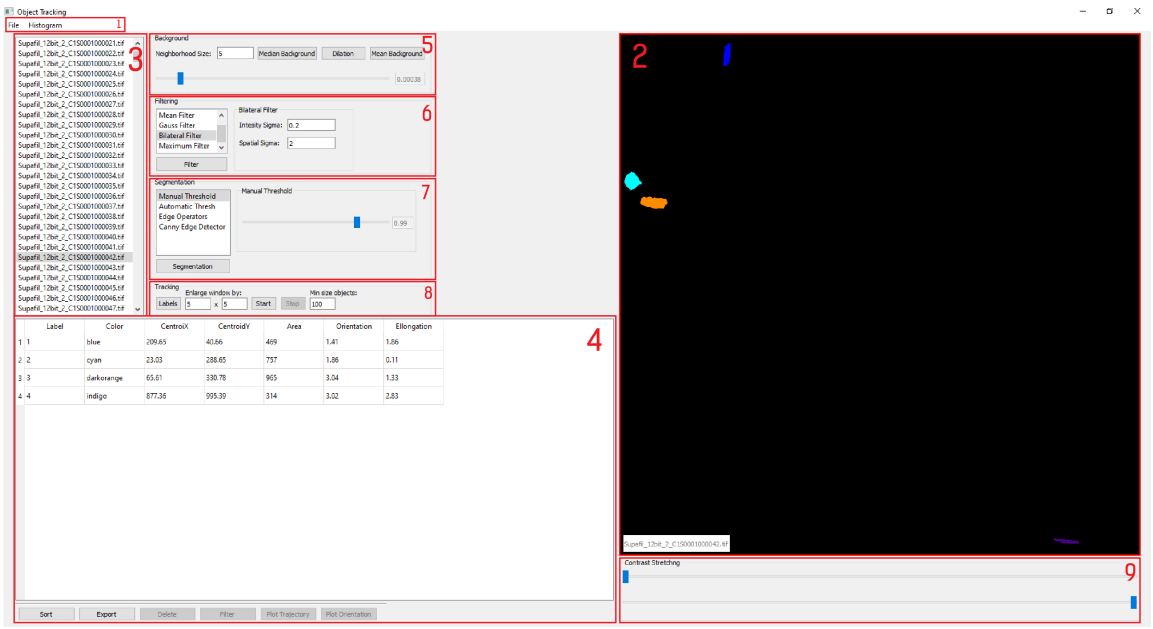
Seznam použitých zkratek

x, y, φ - souřadnice
$\mathbb{R}$ - Reálná čísla
$\mathbb{C}$ - Komplexní čísla
$\mathbb{Z}$ - Celá čísla
$\mathbb{N}$ - Přirozená čísla
$\mathcal{F}$ - Fourierova transformace
$\mathcal{D}$ - Diskrétní Fourierova transformace
$\mathcal{F}^{-1}$ - inverzní Fourierova transformace
$\mathcal{D}^{-1}$ - inverzní diskrétní Fourierova transformace
e - Eulerovo číslo
$\theta(\varphi)$ - interpolační křivka
$\lambda(\gamma, \delta, t)$ - aproximační křivka
$\varphi(\gamma, x)$ - aproximační funkce
$\varrho(X, Y)$ - metrika
$S(x)$ - splajn
F - funkce
$||.||$ - norma
$\mathbf{H}$ - Konvoluční jádro
$f(x, y)$ - spojitá obrazová funkce
$\mathbf{f}(x, y)$ - diskrétní obrazová funkce
$\mathcal{I}_{m \times n}$ - obrazová matice
$\mathcal{P}[i, j], \mathcal{Q}[i, j]$ - pixel
τ - prah
σ^2 - rozptyl
σ - směrodatná odchylka
μ - střední hodnota
G - gradient
G_x - gradient v horizontálním směru
G_y - gradient ve vertikálním směru
T - těžiště
O - objekt
$\bar{x}, \bar{y}$ - souřadnice těžiště
ε - elongace
$\mathcal{B}_M$ - mediánové pozadí
$\mathcal{B}_P$ - průměrové pozadí
$\mathcal{M}$ - maska
$\mathcal{O}^P_\circ$, $\mathcal{O}^P_\square$, $\mathcal{N}^P_4$, $\mathcal{N}^P_8$ - okolí pixelu
r - poloměr okolí
α - orientace objektu
S - strukturní element
$\mathbf{E}$ - eroze
$\mathbf{D}$ - dilatace
$m(p, q)$ - geometrický moment řádu $p + q$
$m_c(p, q)$ - centrální geometrický moment řádu $p + q$
$m_{cn}(p, q)$ - centrální normovaný geometrický moment řádu $p + q$

$m_h(p, q)$ - hlavní geometrický moment řádu $p + q$
$\mathcal{A}$ - plocha objektu

A Manuál k jednoúčelové aplikaci

V této kapitole si přiblížíme ovládaní námi vytvořené aplikace. Její uživatelské prostředí včetně jednotlivých sekcí, které si dále přiblížíme můžeme vidět na obrázku (32).



Obrázek 32: Uživatelské prostředí aplikace

Začneme od nabídky menu, která je označena jako sekce 1.

Sekce 1

V menu můžeme vidět dvě možnosti *File* a *Histogram*. V možnosti File můžeme po rozkliknutí dále nalézt podmožnosti *Open*, *Save image as* a *Close* jak můžeme vidět na obrázku (odkaz na obrázek).

Open První možnost, Open, nám slouží k nahrání obrazů, které budeme chtít dále zpracovávat. Povoleným typem jsou pouze **.tif* obrazy.

Save image as Jak značí název, tato možnost nám slouží k uložení obrazové matice, která se momentálně zobrazuje v sekci 2. Obraz můžeme uložit ve formátech **.jpg*, **.png* a **.tif*.

Close Poslední možnost nám slouží k ukončení aplikace. Po její stisknutí se nám objeví upozornění, zda-li opravdu chceme aplikaci ukončit a možnostmi *Ano* a *Ne*.

Histogram Druhá možnost hlavní menu, Histogram má pouze jednu podmožnost, jak můžeme vidět na obrázku (odkaz na obrázek). Po kliknutí na tuto možnost se nám vykreslí histogram obrazové matice, která je zobrazena v sekci 2 včetně kumulativní histogramu.

Sekce 2

Tato sekce nám slouží k vizualizaci obrazové matice, kterou momentálně zpracováváme.

Sekce 3

V této sekci se nám zobrazují názvy všech obrázků, které jsme nahráli do aplikace pomocí možnosti *Open* z hlavního menu. Pomocí dvojkliknutí je možné mezi jednotlivými obrazy přepínat, kdy při přechodu vždy na nový obraz aplikujeme metody, které jsme do té doby použili, tzn. pokud provedeme filtraci šumu v obraze a přejdeme na jiný obraz, aplikuje se nám tato filtrace námi vybranou metodou i na tento obraz.

Sekce 4

Jak můžeme vidět, v této sekci se nám zobrazují data o jednotlivých vláknech. Při postupném sledování objektů se nám nově nalezené parametry začnou psát na nové řádky. Dále můžeme pod tabulkou vidět šest tlačítek.

Sort Seřadí nám data v tabulce podle hodnoty *Label*. Seřazení je nezbytné pro správné fungování dalších operací s daty.

Export Tato možnost nám umožňuje uložení tabulky ve formátu **.xlsx*.

Delete Slouží k vymazání jednoho či více řádků data z tabulky.

Filter Tímto tlačítkem voláme funkci, která nás upozorní na vlákna, o kterých máme v tabulce méně než dvacet záznamů a nabídne nám možnost data těchto vláken odstranit. Dále provede interpolaci x-ové a y-ové složky těžiště a v případě, že se u některých vláken objeví problémová data, upozorní nás na tuto skutečnost a opět nabídne možnost odstranění záznamů těchto vláken.

Plot Trajectory Pomocí této možnosti se nám vykreslí grafy trajektorií pro všechna vlákna, jejichž data jsme ponechali v tabulce, za předpokladu, že o každém z těchto vláken máme alespoň čtyři záznamy. Pokud tato podmínka není splněna, vypíše se nám upozornění o tom, že graf trajektorie pro tato vlákna nemůže být vykreslen.

Plot Orientation Poslední možnost nám taktéž vykreslí grafy, ale v tomto případě se jedná o grafy orientací a elongací jednotlivých vláken. Platí zde stejná podmínka jako pro vykreslení grafů trajektorie.

Nyní se přesuneme k sekcím, které nám slouží k aplikaci určitých metod na zpracovávaný obraz. Jednotlivé sekce nás navádějí v postupu, to znamená, že se nám postupně zpřístupňují jednotlivé funkce, tak aby byl zachován ideální postup zpracování obrazové matice.

Sekce 5

Tato sekce slouží k aplikaci sekvenční filtrace na obraz. Máme zde opět několik tlačítek a oken, jejichž funkci si nyní popíšeme v pořadí, v kterém se postupně zpřístupňují.

Neighborhood Size V tomto okně si můžeme zvolit velikost parametru p, která jsme si uvedli u definic (2.24) a (2.25). Hodnota tohoto parametru musí být přirozené číslo.

Median Background Tímto tlačítkem vytvoříme mediánové pozadí zpracovávané obrazové matice v závislosti na zadaném parametru p a dále na toto pozadí aplikujeme mediánový filtr, pro snazší segmentaci.

Slider Pomocí slideru si můžeme nastavit hodnotu prahu pro prahování vytvořeného mediánového pozadí, čímž vytvoříme binární masku.

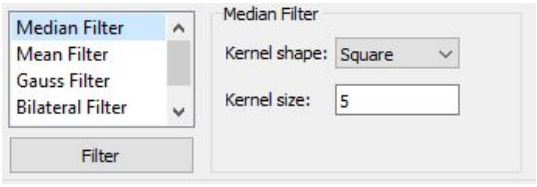
Dilation Tato možnost nám provede dvě morfologické operace, a to nejprve otevření a poté dilataci.

Mean Background Poslední tlačítko nám vytvoří průměrové pozadí v závislosti na parametru p a vytvořené masce. Nakonec se provede vydělení původního obrazu tímto pozadím.

Sekce 6

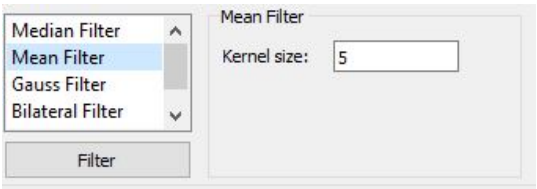
V této sekci dochází k filtraci obrazové matice. Konkrétně máme na výběr z pěti možností, a to *Median Filter*, *Mean Filter*, *Gauss Filter*, *Bilateral Filter* a *Maximum Filter*. Pro výběr požadovaného filtru je potřeba na jeho název v seznamu dvakrát kliknout. Po zvolení filtru se nám objeví příslušné okno, ve kterém můžeme zadat volitelné parametry tohoto filtru. Všechny filtry jsou aplikovány na obrazovou matici, která nám vznikla po stisknutí tlačítka *Mean Background*.

Median Filter První možností, která je automaticky vybrána při přesunu do této sekce, je mediánový filtr. Můžeme zde volit dva parametry, a to tvar okolí, čtvercové nebo kruhové a jeho velikost, jak můžeme vidět na obrázku (33.)



Obrázek 33: Volba parametrů mediánového filtru

Mean Filter Druhou možností je filtr typu dolní propust, u kterého volíme je jeden parametr, jak můžeme vidět na obrázku (34), a tím je velikost konvolučního jádra, které má automaticky čtvercový tvar.

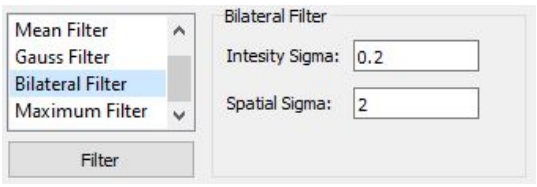


Obrázek 34: Volba parametrů filtru typu dolní propust

Gauss Filter Dále na obraz můžeme aplikovat Gaussův filtr, u kterého opět volíme pouze jeden parametr, jak můžeme vidět na obrázku (35), a tím je σ z rovnice (2.1).

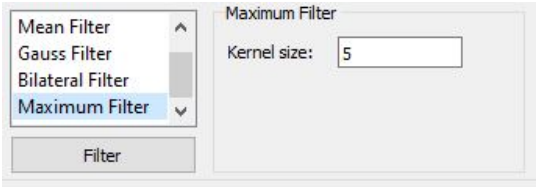
Obrázek 35: Volba parametrů Gaussova filtru

Bilateral Filter Předposlední možností je bilaterální filtr, kde je potřeba zadat hodnoty dvou parametrů, jak můžeme vidět na obrázku (36). Hodnota *Intensity Sigma* odpovídá σ_r a *Spatial Sigma* odpovídá σ_s z definice (2.9).



Obrázek 36: Volba parametrů bilaterálního filtru

Maximum Filter Posledním filtrem, který můžeme na obraz aplikovat, je filtr typu maxima. Zde volíme opět pouze jeden parametr, jak můžeme vidět na obrázku (37), kterým si určujeme velikost s kterým bude tento filtr pracovat a které bude mít vždy čtvercový tvar.



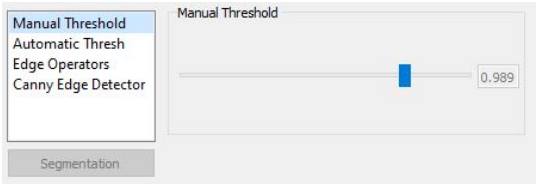
Obrázek 37: Volba parametrů filtru typu maxima

Po vybrání filtru a nastavení parametrů je potřeba stisknout tlačítko *Filter*, které námi zvolený filtr aplikuje.

Sekce 7

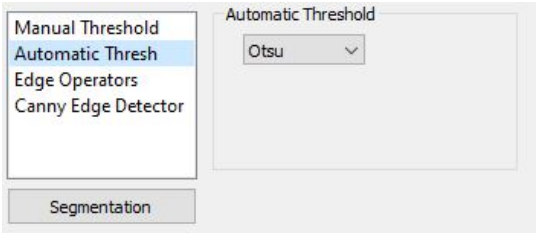
Pomocí této sekce provádíme segmentaci filtrovaného obrazu. Opět máme na výběr z několika možností, konkrétně to pak jsou *Manual Threshold*, *Automatic Thresh*, *Edge Operators* a *Canny Edge Detector*. Stejně jako u filtrů i zde vybereme požadovanou možnost ze seznamu dvojklikem. Po výběru se nám opět objeví příslušné okno, ve kterém můžeme nastavit volitelné parametry.

Manual Threshold První možností je manuální nastavení prahu. Po zvolení se nám zobrazí okno, které můžeme vidět na obrázku (38). Pomocí slideru si můžeme nastavit hodnotu požadovaného prahu. Aktuální hodnotu prahu můžeme vidět v okýnku na pravé straně od slideru.



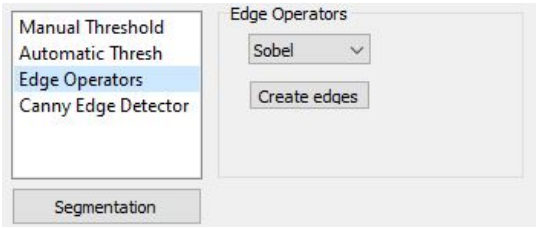
Obrázek 38: Volba parametrů filtru typu maxima

Automatic Thresh Další možností je využití automatických metod hledání prahu. Po rozkliknutí se nám objeví okno, které můžeme vidět na obrázku (39), ve kterém máme možnost vybrat si ze tří algoritmů, a to *Otsuho algoritmus*, *trojúhelníkový algoritmus* a nebo *metodu ISODATA*. Pro aplikování námi vybraného algoritmu je potřeba kliknout na tlačítko *Segmentation* pod seznamem segmentačních možností.



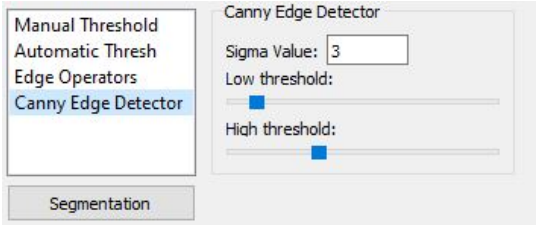
Obrázek 39: Volba parametrů filtru typu maxima

Edge Operators Po rozkliknutí této možnosti si můžeme vybrat operátor pro hledání hran v obrazové matici, jak můžeme vidět na obrázku (40). Pro nalezení hran zvoleným operátorem je potřeba kliknout na tlačítko *Create edges*. Pomocí tlačítka *Segmentation* je poté vyplněním těchto hran vytvořen binární obraz.



Obrázek 40: Volba parametrů filtru typu maxima

Canny Edge Detector Poslední možností je Cannyho hranový detektor, u kterého volíme tři parametry jak jde vidět na obrázku (41). Těmito parametry jsou postupně σ, pro aplikaci Gaussova Filtru, a poté hodnoty dolního a horního prahu hystereze, které volíme pomocí dvou sliderů. Kliknutím na tlačítko *Segmentation* aplikujeme morfologickou operaci uzavření, tak abychom částečně propojili některé nalezené hrany.



Obrázek 41: Volba parametrů filtru typu maxima

Sekce 8

Předposlední sekce je zaměřená na sledování objektů.

Labels Pomocí tohoto tlačítka aplikujeme labeling na obrazovou matici, vytvořenou některou ze segmentačních metod.

Enlarge window by U této možnosti mámě dvě okna. Hodnot, které zadáme do těchto oken, nám určují o kolik pixelů bude zvětšeno okno, které používáme v prvním kroku sledování objektů, který byl popsán v kapitole (3.3). Hodnota v prvním okně nám rozšiřuje okno v horizontálním směru a druhá ve vertikálním. Hodnoty musí být nezáporná celá čísla.

Start Spouští samotné sledování objektů tak, jak bylo popsáno v kapitole (3.3).

Stop Ukončí algoritmus sledování objektů.

Min size objects

Sekce 9

Poslední sekce nám slouží ke zvýšení kontrastu obrazové matice. Nalézají se zde dva slidery, kterými určujeme hodnoty $[z_1, z_k]$ z kapitoly (2.2). V případě, že je v Sekci 2 momentálně zobrazen obraz po filtraci, upravujeme kontrast u takovéhoto obrazu, v jakémkoli jiném případě se nám zobrazí vstupní obrazová matice s upraveným kontrastem.