UNIVERZITA PALACKÉHO V OLOMOUCI
PŘÍRODOVĚDECKÁ FAKULTA
KATEDRA MATEMATICKÉ ANALÝZY A APLIKACÍ MATEMATIKY

DIPLOMOVÁ PRÁCE

Metody segmentace obrazu

Katedra matematické analýzy a aplikací matematiky
Vedoucí diplomové práce: **RNDr. Tomáš Fürst, Ph.D.**
Vypracoval: **Bc. Jana Falaštová**
Studijní program: N1103 Aplikovaná matematika
Studijní obor: Matematické a počítačové modelování
Forma studia: prezenční
Rok odevzdání: 2015

BIBLIOGRAFICKÉ IDENTIFIKACE

Autor: Bc. Jana Falaštová

Název práce: Metody segmentace obrazu

Typ práce: Diplomová práce

Pracoviště: Katedra matematické analýzy a aplikací matematiky

Vedoucí práce: RNDr. Tomáš Fürst, Ph.D.

Rok obhajoby práce: 2015

Abstrakt: Práce se zabývá metodami segmentace obrazu. Nejdříve je uvedena metoda prahování a postupy pro volbu vhodného prahu. V další části je uvedena metoda aktivních kontur Chan-Vese. V práci je odvození metody a následný přechod k numerickému modelu. Dále je popsán vlastní algoritmus programu, který detekuje jednotlivá obarvená jádra buněk a rozděluje shluky na jednotlivá jádra. V další kapitole jsou uvedeny výsledky metody prahování a metody aktivních kontur s různými vstupními parametry. V poslední části je uveden popis dalšího programu a následné srovnání obou programů na konkrétních snímcích.

Klíčová slova: Segmentace obrazu, prahování, metoda aktivních kontur

Počet stran: 56

Počet příloh: 1

Jazyk: česky

BIBLIOGRAPHICAL IDENTIFICATION

Author: Bc. Jana Falaštová

Title: Image segmentation methods

Type of thesis: Master's

Department: Department of Mathematical Analysis and Application of Mathematics

Supervisor: RNDr. Tomáš Fürst, Ph.D.

The year of presentation: 2015

Abstract: The Thesis delas with image segmentation methods. First, thresholding is introduced and various techniques to select the optimal threshold are discussed. Next, the Chan-Vese method of active contour is introduced. The Chan-Vese method is used to detect DAPI colored nuclei of individual cells. An algorithm for an automatic detection of the nuclei is presented. The influence of various parameters on the active contour algorithm is discussed. The algorithm is used on a test set of images, each containing several nuclei, usually in close contact with one another. The performance of the proposed algorithm is tested and compared to other methods of image segmentation.

Key words: Image segmentation, active contour, threshold

Number of pages: 56

Number of appendices: 1

Language: Czech

Prohlášení

Prohlašuji, že jsem diplomovou práci zpracovala samostatně pod vedením Tomáše Fürsta s použitím uvedené literatury.

V Olomouci dne Jana Falaštová

Obsah

Úvod **7**

1 Základní pojmy **10**
1.1 Reprezentace obrazů v Matlabu 10
1.2 Další pojmy 11

2 Prahování **13**
2.1 Základní iterační metoda 14
2.2 Metoda prohlubní 15
2.3 Otsuova metoda 15
2.4 Lokální metody 18

3 Metoda aktivních kontur **19**
3.1 Metoda aktivních kontur Chan - Vese 19

4 Program využívající metodu aktivních kontur **26**
4.1 Popis postupu 26
4.2 Parametry 28
4.3 Algoritmus 30
4.4 Proces rozdělení 31
4.5 Rozdělení pomocí rozšíření oblasti 33
4.6 Analýza obrazu v Matlabu 33

5 Výsledky **37**
5.1 Prahování 37
5.2 Metoda aktivních kontur 39

6 Srovnání **42**
6.1 Popis programu založeného na prohledávání hranic objektů 42
6.2 Nevýhody jednotlivých postupů 43
6.3 Srovnání výsledků 46
6.4 Postup využívající metodu watershed 49

Závěr **53**

Literatura **54**

Přílohy **56**

Úvod

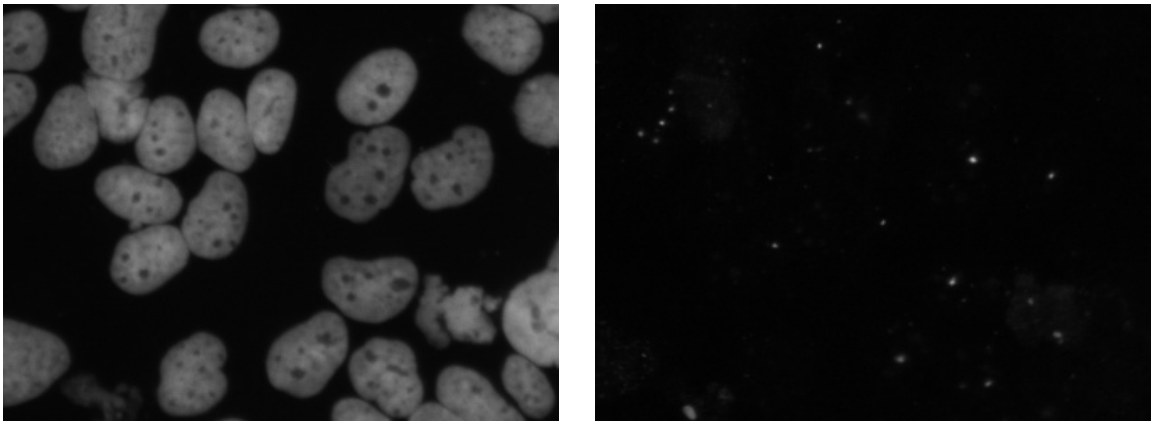
Úkolem segmentace obrazu je rozdělit snímek na disjunktní oblasti, jejichž body mají nějakou společnou vlastnost (barvu, jas, strukturu). Metod je mnoho a pracují na různých principech. Jsou to metody založené na detekci hran (Cannyho hranový detektor [1]), regionově zaměřené metody, metody využívající histogramu (prahování [2]) nebo metody založené na parciálních diferenciálních rovnicích (metoda aktivních kontur [3], [4]). V práci jsem se zaměřila na metodu prahování (threshold) a volbu vhodného prahu. Tato metoda je jedna z nejjednodušších, ale také nejpoužívanějších jelikož převádí snímky na binární. Binární snímky získané prahováním se často používají jako vstupní snímky pro jiné metody. Hlavní metodou, kterou jsem se zabývala ve své práci, je metoda aktivních kontur (active contour). Variant této metody je mnoho. První metoda aktivních kontur byla uvedena v [3].

Metody segmentace obrazu se využívají v mnoha odvětví, jako je optická kontrola kvality, detekce objektů a mnoho využití nachází především v lékařství.

Každá buňka v lidském těle obsahuje jádro. Jádro obsahuje chromozomy, které jsou tvořeny dlouhými řetězci DNA. Integrita DNA je stále ohrožována jak vnitřními vlivy (metabolismem) tak vnějšími vlivy. Vnější vlivy mohou být například UV záření, rentgenové záření, gama záření, chemikálie v prostředí nebo viry. Poškozením DNA může být mutace nebo rozpad řetězce. Poškození může způsobit mnoho závažných nemocí jako je nemoc z ozáření, rakovina nebo předčasné stárnutí. Buňka má určité mechanismy na opravu poškození a většinu poškození umí opravit (viz [5]). Pokud je poškození příliš velké, může dojít k sebezničení buňky (apoptóza) nebo imunitní systém vyhodnotí tyto buňky jako abnormální a zničí je. V některých případech ale k opravě nedojde a dojde k nekontrolovatelnému množení a dělení buněk s poškozenou DNA, které je příčinou rakoviny (viz [6]). Buňky obsahují několik proteinů, které zajišťují opravení rozpadu DNA. Tyto mechanizmy oprav chyb DNA mohou být klíčem k lékům na rakovinu a proto jsou intenzivně zkoumány.

Při zkoumání integrity genomu se vzorek ozáří laserem, rentgenovým zářením nebo se aplikují nějaké genotoxické látky. Tím dojde k poškození DNA a následně se pozorují reakce buněk. Na poškození DNA se váže skupina proteinů, které mají funkci opravovat poškození. Vzorek se obarví DAPI, což je chemická fluorescenční látka, která se dobře váže na DNA. Po navázání s DNA zvyšuje svoji intenzitu fluorescence. Dále se na vzorek aplikuje jiná látka, která se naváže na proteiny a po navázání se obarví jinou barvou než DNA obarvena DAPI. K analýze máme vždy dva snímky daného vzorku

získané laserovým rastrovacím mikroskopem [7]. První obsahuje pouze obarvená jádra buněk, jako na Obrázku 0.1 (a). Druhý snímek je snímek stejného místa ve stejném čase pořízen v jiné vlnové délce. Na tomto snímku již nejsou vidět jádra, ale tzv. fokusy, jako na Obrázku 0.1 (b). To jsou místa, kde došlo ke zlomu DNA a na těchto místech jsou nahromaděny proteiny, které fluoreskují. Cílem analýzy je kvantifikovat počet fokusů na jedno jádro. K tomu je potřeba umět rozdělit shluky jader na jednotlivá jádra. Shluky navíc nejsou ve vzorcích ojedinělé jelikož buňky mají tendenci se shlukovat. Pokud by nedošlo ke správnému rozdělení, byla by hodnota poškození DNA uměle nadhodnocena. Tímto problémem se zabývá rozsáhlý výzkum integrity genomu, který probíhá na Ústavu molekulární a translační medicíny Lékařské fakulty Univerzity Palackého v Olomouci [8].



(a) (b)

Obrázek 0.1: (a) snímek s obarvenými jádra, (b) snímek s fokusy

Hlavním cílem této práce je implementovat programový kód v Matlabu, který detekuje jednotlivá obarvená jádra a rozdělí jejich shluky. Dalšími cíli práce je pochopit a popsat metody prahování a metodu aktivních kontur. Dále prozkoumat matematické pozadí metody aktivních kontur. Dalším cílem je ukázat výsledky programu na testovací sadě snímků získaných z Laboratoře integrity genomu.

V první kapitole práce se zaměřím na teoretické poznatky spojeny se segmentací obrazu. Ve druhé kapitole popíši metodu prahování a několik způsobů volby vhodného prahu. Ve třetí kapitole se budu zabývat metodou aktivních kontur a především metodou Chan-Vese [4], jelikož tuto metodu využívám ve svém programu.

Ve čtvrté kapitole se zaměřím na praktickou část. Popíši algoritmus programu, který jsem napsala v programu Matlab. V páté kapitole ukáži výsledky obou metod, prahování a metody aktivních kontur, při různé volbě parametrů a nastavení.

V poslední části stručně popíši program vytvořený ve spolupráci s Tomášem Fürstem, který je založený na prohlížení hranice objektů. Ukáži nedostatky tohoto i prvního programu založeného na metodě aktivních kontur. Nakonec srovnám výsledky obou metod na testovací sadě snímků.

1 Základní pojmy

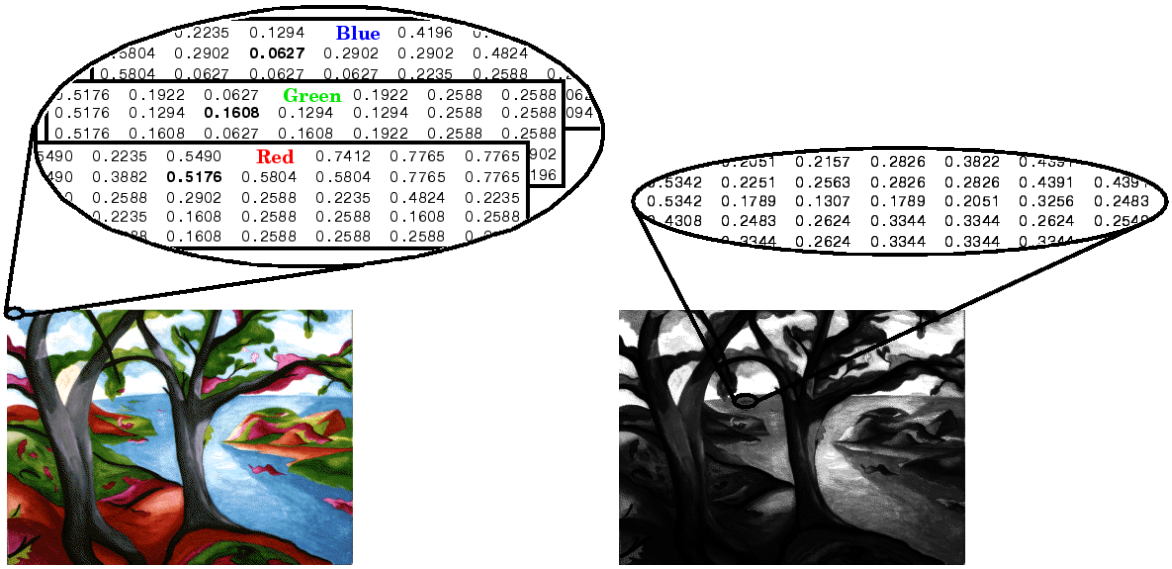
1.1 Reprezentace obrazů v Matlabu

Při práci s obrazy v počítači musíme pracovat s diskrétními daty. Obrazy jsou proto vyjádřeny maticemi a můžeme s nimi i takto pracovat. Vyjádření může být několika způsoby.

RGB obraz je barevný obraz reprezentovaný polem $m \times n \times 3$, které definuje červenou (R), zelenou (G) a modrou (B) barvu pro každý pixel (Obrázek 1.1 (a))

Obraz ve stupních šedé je reprezentován maticí velikosti $m \times n$ intenzit jasu jednotlivých pixelů (Obrázek 1.1 (b))

Binární obraz je černobílý obraz vyjádřen maticí $m \times n$, jejichž prvky mohou nabývat pouze dvou hodnot, a to buď 0 (černá) nebo 1 (bílá).



(a) (b)

Obrázek 1.1: (a) reprezentace RGB obrazu, (b) reprezentace obrazu ve stupních šedé. (zdroj [18])

Hodnoty u RGB obrazů a obrazů ve stupních šedé mohou být z různých rozmezí, tedy různého typu

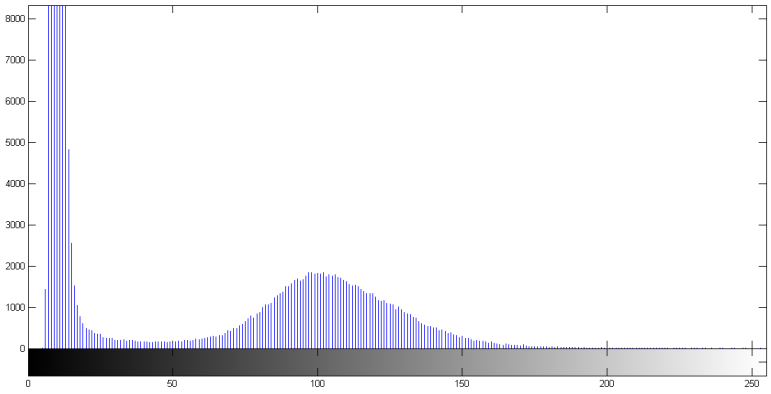
double	hodnoty v rozmezí [0; 1]
uint8	8-bitový formát, tj. celočíselné hodnoty v rozmezí [0; 255]

uint16 16-bitový formát, tj. celočíselné hodnoty v rozmezí [0; 65535]

Histogram obrazu

Histogram obrazu ve stupních šedé znázorňuje zastoupení pixelů s různým jasem. Na ose x jsou jednotlivé složky jasu a osa y představuje počet pixelů. Je to tedy sloupcový graf, kde výška k-tého sloupce udává počet pixelů v obraze s k-tým stupněm šedé.

Bi-modální histogram je takový, který má výrazné dva vrcholy (Obrázek 1.2) . Tento histogram znázorňuje, že na obraze jsou dva druhy pixelů, například tmavé pozadí se světlými objekty.



Obrázek 1.2: Bi-modální histogram

1.2 Další pojmy

Definice 1.1. Heavisideova funkce (nebo také jednotkový skok) H je nespojitá funkce definovaná předpisem

$$H(x) = \begin{cases} 0 & \text{pro } x < 0 \\ 1 & \text{pro } x \geq 0 \end{cases}$$

Definice 1.2. Diracova míra (nazývaná také δ-funkce) δ je formálně definovaná předpisem

$$\delta(x) = \begin{cases} +\infty & \text{pro } x = 0 \\ 0 & \text{pro } x \neq 0 \end{cases}$$

Tvrzení 1.3. *Pro Heavisideovu funkci a Diracovu δ-funkci platí následující vzájemné vztahy:*

1. *Heavisideova funkce je integrál Diracovy δ-funkce*

$$H(x) = \int_{-\infty}^{x} \delta(s)\, ds$$

2. *Derivace Heavisideovy funkce je δ-funkce*

$$\frac{dH(x)}{dx} = \delta(x)$$

3. *Integrál δ-funkce přes celou reálnou osu je 1*

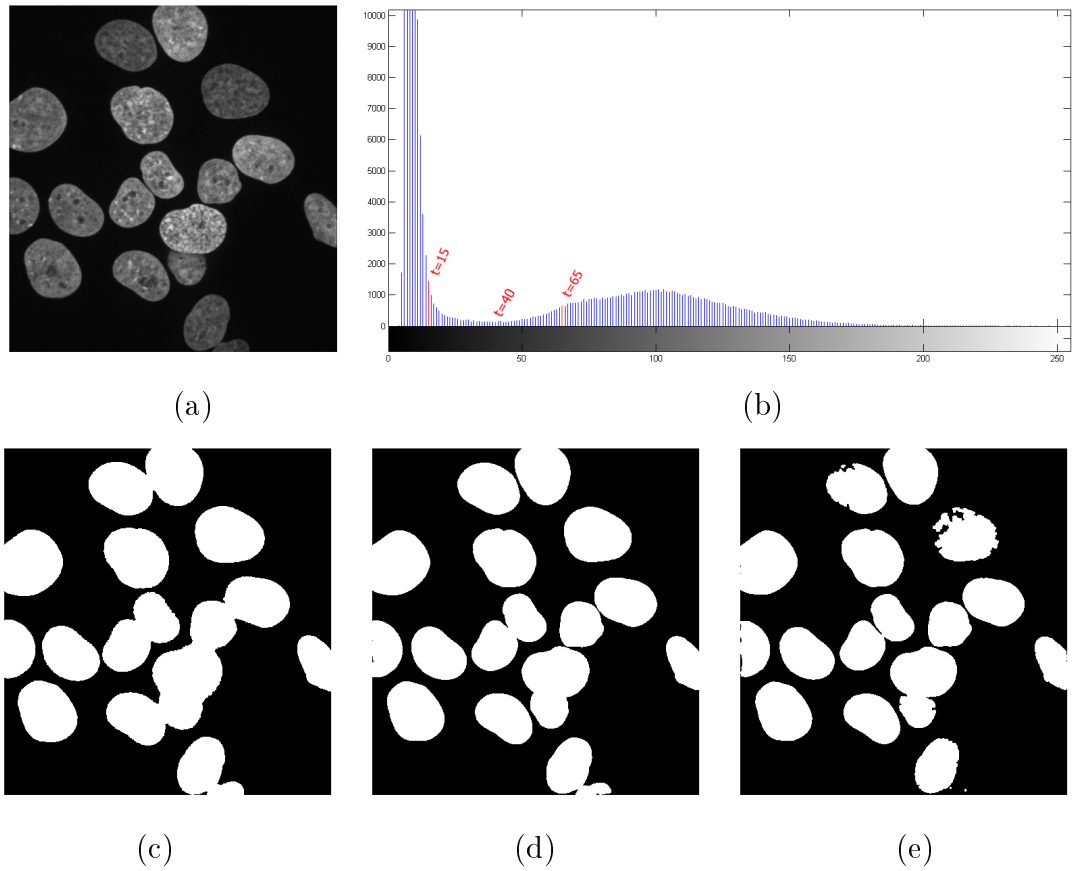
$$\int_{-\infty}^{+\infty} \delta(x)\, dx = 1$$

Poznámka 1.4. Jelikož δ-funkce není funkce ale distribuce neboli zobecněná funkce, proto předchozí vztahy nebereme v klasickém vztahu, ale ve smyslu distribucí. Více o distribucích a operacích s nimi nalezneme například na [9].

2 Prahování

Jednou z nejjednodušších metod segmentace obrazu je prahování (angl. threshold). Používá se především v situacích, kdy víme, že pozadí a objekty mají rozdílnou barvu nebo jas. Metoda převádí obrázek ve stupních šedi na binární obrázek. Pro tuto metodu musíme zadat hodnotu prahu, kterou si zvolíme, nebo ji můžeme získat pomocí nějaké automatické metody. Metoda převádí pixely s hodnotou jasu menší než je práh na nulu, tedy černou, pixely s hodnotou větší než práh na jedničku, tedy bílou.

$$T(x,y) = \begin{cases} 1 & \text{pro } f(x,y) \geq t \\ 0 & \text{pro } f(x,y) < t \end{cases}$$



Obrázek 2.1: Prahování s různými prahy: (a) originální obraz ve stupních šedé ve formátu uint8, (b) histogram originálního obrazu se zvýrazněnými hodnotami prahů, (c) obraz po prahování s prahem $t_1 = 15$, (d) obraz po prahování s prahem $t_2 = 40$, (e) obraz po prahování s prahem $t_3 = 65$

Správná volba prahu je velice důležitá, jak můžeme vidět na Obrázku 2.1. Pokud zvolíme hodnotu prahu nízkou, tak se často stává, že se několik objektů „slije“ do jednoho, jako na Obrázku 2.1 (c). Pokud zvolíme hodnotu prahu vysokou, tak získáme lépe rozdělené objekty, ale problém je, že tmavší objekty nebývají po prahování celé, jako na Obrázku 2.1 (e). Proto musíme nalézt optimální hodnotu, tak abychom těmto dvěma negativním vlivům předešli.

Automatických metod pro nalezení prahu je několik. Dvě nejjednodušší jsou metody, kdy hodnotu prahu vezmeme jako střední hodnotu nebo medián hodnot histogramu. Přehled metod prahování můžeme naleznout například v [10]. Několik metod si popíšeme.

2.1 Základní iterační metoda

Tuto metodu nalezneme v [11]. Jedná se o jednoduchou iterační metodu, která využívá hodnot histogramu. Postup je následující :

1. Zvolíme počáteční hodnotu t_0, která může být libovolná, např. polovina z rozsahu snímku.

2. Pixely obrazu rozdělíme na dvě skupiny

$$G_1 = \{f(x,y) : f(x,y) > t\} \text{ (pixely objektu)}$$
$$G_2 = \{f(x,y) : f(x,y) \leq t\} \text{ (pixely pozadí)}$$

3. Vypočteme průměrnou hodnotu pro každou skupinu

$$m_1 = \text{průměr hodnot } G_1$$
$$m_2 = \text{průměr hodnot } G_2$$

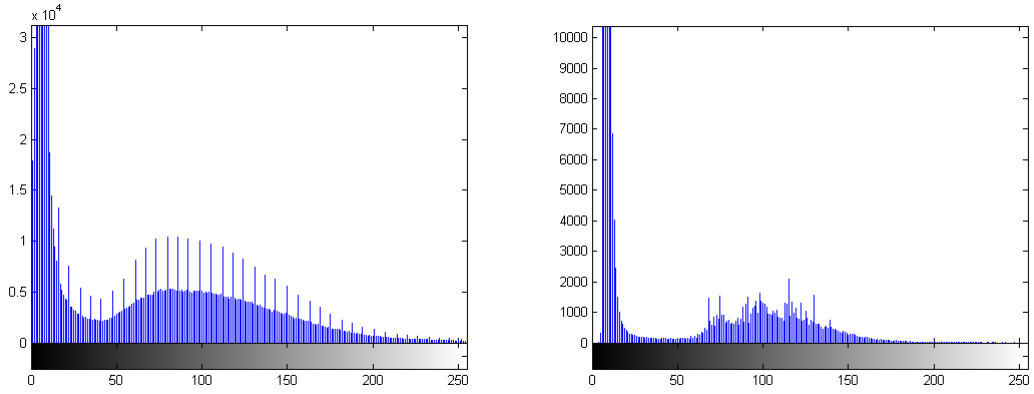
4. Vypočteme novou hodnotu prahu

$$t' = \frac{m_1 + m_2}{2}$$

5. Vrátíme se zpět ke kroku 2. a postup opakujeme pro novou hodnotu prahu dokud nebude platit $\mid t - t' \mid < \varepsilon$, kde $\varepsilon > 0$ dostatečně malé.

2.2 Metoda prohlubní

Metoda má využití především na snímcích s bi- nebo multi-modálním histogramem. Vezmeme-li případ s bi-modálním histogramem, ten má prohlubeň mezi dvěma výraznými vrcholy. Hledáme tedy lokální maxima a mezi nimi hledáme lokální minimum. Hodnotu optimálního prahu zvolíme jako bod tohoto minima. Problém u této metody nastává, pokud histogram není hladký, ale je tzv. chlupatý (Obrázek 2.2 (a)) nebo s několika lokálními extrémy (Obrázek 2.2 (b)). První případ je způsoben například nepřesným převodem na jiný datový typ snímku. Pro takové histogramy nebývá snadné nalézt extrémy nebo jich je několik. Před hledáním extrému můžeme provést vyhlazení histogramu nebo jej aproximovat křivkou. Pokud budeme histogram prokládat křivkou je nutné si rozmyslet jakým typem křivky. Pokud máme soubor snímků s podobnými histogramy, můžeme všechny histogramy prokládat stejným tipem křivky, který nám poskytne dobrou aproximaci. Pokud máme ovšem soubor se snímky s různými rozděleními histogramu, tak proložení některých histogramů bude s příliš velkou chybou. Pokud budeme hledat správný typ křivky pro každý snímek zvlášť, bude to příliš časově náročné. Bi-modální histogramy se nejčastěji prokládají kombinací dvou Gaussových křivek.



(a) (b)

Obrázek 2.2: Histogramy: (a) chlupatý histogram, (b) histogram s více lokálními extrémy

2.3 Otsuova metoda

Otsuova metoda je jedna z nejznámějších a nejpoužívanějších metod prahování. Metoda nejlépe funguje na obrazy s bimodálním histogramem.

Máme obraz s L stupni šedi $(0, 1, 2, ..., L-1)$. Počet pixelů s hodnotou jasu i

označme n_i a celkový počet pixelů je $N = n_0 + n_1 + \cdots + n_{L-1}$. Relativní četnosti i-tého jasu označme:

$$p_i = \frac{n_i}{N}$$

Předpokládáme, že chceme pixely rozdělit do dvou skupin $C_1 = \{0, 1, \ldots, t\}$ (pozadí) a $C_2 = \{t+1, t+2, \ldots L-1\}$ (objekty), kde t je práh. Pokud si označíme počet pixelů v jednotlivých skupinách

$$N_1 = \sum_{i=0}^{t} n_i$$
$$N_2 = \sum_{i=t+1}^{L-1} n_i$$

Poměry zastoupení jednotlivých skupin v celém obrazu jsou dány

$$\omega_1(t) = \frac{N_1}{N}$$
$$\omega_2(t) = \frac{N_2}{N}$$

Pokud si navíc označíme relativní četnosti i-tého jasu v jednotlivých skupinách

$$p_i^1 = \frac{n_i}{N_1}$$
$$p_i^2 = \frac{n_i}{N_2}$$

vypočteme střední hodnoty stupňů šedi obsažených v jednotlivých skupinách

$$\mu_1(t) = \sum_{i=0}^{t} i\, p_i^{C_1}$$
$$\mu_2(t) = \sum_{i=t+1}^{L-1} i\, p_i^{C_2}$$

Rozptyly hodnot stupňů šedi obsažených v jednotlivých skupinách vypočteme ze vztahů

$$\sigma_1^2(t) = \sum_{i=0}^{t} (i-\mu_1)^2 p_i^{C_1}$$
$$\sigma_2^2(t) = \sum_{i=t+1}^{L-1} (i-\mu_2)^2 p_i^{C_2}$$

Optimální hodnotu prahu t^* vypočteme jako bod minima variability uvnitř shluků (intra-class variance)

$$\sigma_W^2(t) = \omega_1(t)\,\sigma_1^2(t) + \omega_2(t)\,\sigma_2^2(t)$$

Je ukázáno v [12], že minimalizace variability uvnitř shluků je ekvivalentní s maximalizací variability mezi shluky (inter-class variance)

$$\sigma_B^2(t) = \omega_1(t)\,(\mu_1(t)-\mu)^2 + \omega_2(t)\,(\mu_2(t)-\mu)^2 = \omega_1(t)\,\omega_2(t)\,(\mu_1(t)-\mu_2(t))^2$$

kde $\mu = \sum_{i=0}^{L-1} i p_i$.

2.3.1 Víceprahová Otsuova metoda

Předchozí vztahy pro klasickou Otsuovu metodu lze rozšířit na víceprahové prahování (viz [13]). Hledáme tedy $M-1$ prahů, $\{t_1, t_2, \ldots, t_{M-1}\}$, které rozdělí histogram na M skupin $C_1 = \{0, 1, \ldots, t_1\}, \ldots, C_i = \{t_{i-1}+1, \ldots, t_i\}, \ldots, C_M = \{t_{M-1}+1, \ldots, L-1\}$. Označme počet pixelů v jednotlivých skupinách

$$N_k = \sum_{i \in C_k} n_i \qquad\qquad k = 1, 2, \ldots M$$

Poměry zastoupení jednotlivých skupin v obrazu vyjádříme jako

$$\omega_k = \frac{N_k}{N} \qquad\qquad k = 1, 2, \ldots M$$

Střední hodnotu v jednotlivých skupinách vypočteme ze vztahu

$$\mu_k = \sum_{i \in C_k} i \frac{n_i}{N_k} \qquad\qquad k = 1, 2, \ldots M$$

a střední hodnotu celého obrazu vypočteme

$$\mu = \sum_{i=0}^{L-1} i \frac{n_i}{N}$$

Optimální prahy získáme ze vztahu

$$\left\{t_1^*, t_2^*, \ldots, t_{M-1}^*\right\} = \max\left\{\sigma_B^2\left(t_1, t_2, \ldots, t_{M-1}\right)\right\}$$

kde $\sigma_B^2 = \sum_{k=1}^{M} \omega_k \left(\mu_k - \mu\right)^2$

2.4 Lokální metody

Pro některé obrazy nelze použít jednu prahovací hodnotu na celý obraz. Pokud je například scéna nasvícena z jednoho směru, dostaneme po prahování jen jednu stranu objektu. Pokud jsou některé objekty přesvícené nepříznivě ovlivňují výpočet optimálního prahu, který může být posunut. V těchto případech je vhodné použít lokální metody prahování, kdy se určité metody použijí na menší oblasti v obraze. Jednotlivé části obrazu mohou mít jinou hodnotu prahu. V případě přesvícených objektů se lokální prahování na menších oblastech provede například ve dvou krocích, kdy v prvním kroku se vyřadí přesvícené objekty a v druhém kroku získáme ostatní objekty.

3 Metoda aktivních kontur

Metoda aktivních kontur je také známá jako "snake". Metoda je založena na minimalizaci funkcionálu a první model byl uveden v [3]. Její přístup je založen na deformaci počáteční křivky C_1 směrem k hranici objektu, který má být nalezen. Výsledná křivka je taková, že minimalizuje daný funkcionál.

Nejdříve si krátce popíšeme klasický model, který vychází z [3]. Nechť Ω je ohraničená otevřená podmnožina v $\mathbb{R}^2$ s hranicí $\partial\Omega$. Nechť $u : \overline{\Omega} \to \mathbb{R}$ je daný obrázek a $C(s) : [0,1] \to \mathbb{R}^2$ je parametrizovaná křivka. Počáteční křivka C se postupně deformuje. Výsledná křivka minimalizuje (lokálně) funkcionál $J(C)$ daný předpisem

$$J(C) = \alpha \int_0^1 \mid C'(s) \mid^2 ds + \beta \int_0^1 \mid C''(s) \mid ds - \lambda \int_0^1 \mid \bigtriangledown u(C(s)) \mid^2 ds$$

.

Kde α, β a λ jsou kladné parametry. Funkcionál energie se skládá ze dvou částí, vnitřní a vnější energie. První dva výrazy představují vnitřní energii a kontrolují délku a hladkost křivky. Třetí výraz představuje vnější energii, která zajišťuje, že se křivka přitahuje k hranici objektu.

Existuje mnoho dalších variant této metody, odlišují se pouze ve tvaru funkcionálu. Klasický model se řadí mezi tzv. metody aktivních kontur s hranami, jelikož zde vystupuje člen s ∇u. Pokud chceme minimalizovat funkcionál J snažíme se maximalizovat třetí člen $\int_0^1 \mid \bigtriangledown u(C(s)) \mid^2 ds$. Jelikož vysoké hodnoty gradientu u představují hrany v obraze, výsledná křivka se bude přitahovat k hranám. Druhý typ jsou metody aktivních kontur bez hran, kdy ve funkcionálu již nevystupuje člen s gradientem funkce u, ale funkcionál v těchto metodách je speciálním případem Mumford-Shah funkcionálu [14] pro segmentaci obrazů.

3.1 Metoda aktivních kontur Chan - Vese

Metoda byla uvedena v [4], řadí se mezi metody aktivních kontur bez hran a pod tímto názvem ji můžeme také nalézt.

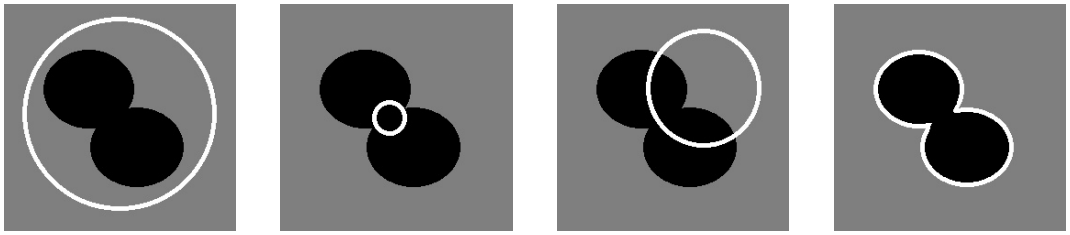
3.1.1 Odvození metody

Vezmeme si binární obrázek u s jedním objektem, který má odlišnou barvu od pozadí. Tedy u je po částech konstantní s hodnotami u^i a u^o. Objekt, který chceme nalézt,

je reprezentován oblastí s hodnotou u^i, jeho hranice označme C_O. Tedy máme $u = u^i$ uvnitř objektu a $u = u^o$ vně objektu. Dále označme oblast ohraničenou křivkou C jako ω, oblast vně křivky C označme $\Omega \setminus \omega$. Vezmeme si potenciál

$$F_1(C) + F_2(C) = \int_\omega |u(x,y) - c_1|^2 dx\, dy + \int_{\Omega\setminus\omega} |u(x,y) - c_2|^2 dx\, dy$$

kde c_1, c_2 jsou závislé na křivce C. c_1 je průměr hodnot u v oblasti ω a c_2 je průměr hodnot u v $\Omega \setminus \omega$.



$F_1(C) > 0$	$F_1(C) = 0$	$F_1(C) > 0$	$F_1(C) = 0$
$F_2(C) = 0$	$F_2(C) > 0$	$F_2(C) > 0$	$F_2(C) = 0$

Obrázek 3.1: Možné případy hodnot výrazů F_1 a F_2 v závislosti na poloze křivky C vzhledem k objektu

Pokud je křivka C vně objektu, poté $F_1(C) > 0$ a $F_2(C) = 0$. Pokud je křivka C uvnitř objektu, potom $F_1(C) = 0$ ale $F_2(C) > 0$. Pokud křivka zasahuje částí do objektu a částí do pozadí poté $F_1(C) > 0$ a $F_2(C) > 0$. Je tedy jasné, že potenciál má minimum na hranici objektu tj. pokud $C = C_O$. Hranice objektu C_O minimalizuje potenciál $F_1 + F_2$.

$$\inf_C \{F_1(C) + F_2(C)\} \approx 0 \approx F_1(C_O) + F_2(C_O)$$

Tento funkcionál nám představuje vnější energii. Stejně jako v klasickém modelu přidáme výrazy představující vnitřní energie. Přidáním výrazů pro omezení délky křivky C a velikosti oblasti ω, dostáváme funkcionál $F(c_1, c_2, C)$ definovaný

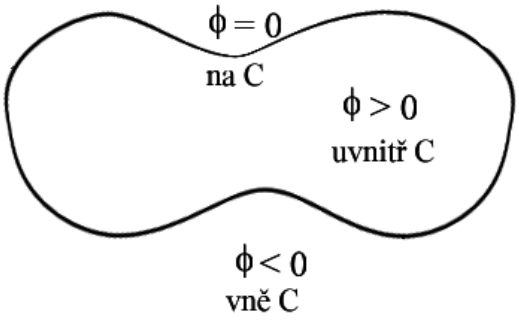
$$\begin{aligned} F(c_1, c_2, C) = {} & \mu \cdot Length(C) + \nu \cdot Area(\omega) + \\ & + \lambda_1 \int_\omega |u(x,y) - c_1|^2 dx\, dy + \lambda_2 \int_{\Omega\setminus\omega} |u(x,y) - c_2|^2 dx\, dy \end{aligned} \tag{3.1}$$

kde $\mu \geq 0$, $\nu \geq 0$, $\lambda_1, \lambda_2 > 0$ jsou dané parametry. (V numerických výpočtech v [4] používají nastavení parametrů $\lambda_1 = \lambda_2 = 1$ a $\nu = 0$.)

3.1.2 Formulace pomocí vodorovných řezů

K řešení vyvíjejících se křivek se nejčastěji používá metoda vodorovných řezů (level set formulation). V této metodě křivku C a oblasti uvnitř ω a vně $\Omega \setminus \overline{\omega}$ křivky C reprezentujeme pomocí lipschitzovské funkce $\phi : \Omega \to \mathbb{R}$:

$$\begin{aligned} \{(x,y) \in \Omega : \phi(x,y) = 0\} &= C = \partial\omega \\ \{(x,y) \in \Omega : \phi(x,y) > 0\} &= \omega \\ \{(x,y) \in \Omega : \phi(x,y) < 0\} &= \Omega \setminus \overline{\omega} \end{aligned} \tag{3.2}$$



Obrázek 3.2: Znázornění hodnot funkce ϕ v závislosti na křivce C

Pomocí této funkce a znalosti Heavisidovy funkce a Diracovy míry vyjádříme části funkcionálu F následovně:

$$Length\,(\phi = 0) = \int_\Omega \delta(\phi(x,y)) \mid \nabla\phi(x,y) \mid \mathrm{d}x\,\mathrm{d}y \tag{3.3}$$

$$Area\,(\phi \geq 0) = \int_\Omega H(\phi(x,y))\,\mathrm{d}x\,\mathrm{d}y \tag{3.4}$$

$$\int_{\phi>0} \mid u(x,y) - c_1 \mid^2 \mathrm{d}x\,\mathrm{d}y = \int_\Omega \mid u(x,y) - c_1 \mid^2 H(\phi(x,y))\,\mathrm{d}x\,\mathrm{d}y \tag{3.5}$$

$$\int_{\phi<0} \mid u(x,y) - c_2 \mid^2 \mathrm{d}x\,\mathrm{d}y = \int_\Omega \mid u(x,y) - c_2 \mid^2 (1 - H(\phi(x,y)))\,\mathrm{d}x\,\mathrm{d}y \tag{3.6}$$

Poznámka 3.1. Správnost formule (3.3) pro délku křivky nemusí být na první pohled jasná. Ukážeme si tedy její možné ověření, které najdeme v [15].

Mějme ϕ definované vztahem (3.2). Nahradíme δ-funkci funkcí

$$\delta_N(t) = \begin{cases} N & \mid t \mid \leq \frac{1}{2N} \\ 0 & \mid t \mid > \frac{1}{2N} \end{cases}$$

kde $N \gg 1$. Poté

$$\int_\Omega \delta(\phi(x,y)) \mid \nabla\phi(x,y) \mid \mathrm{d}x\,\mathrm{d}y \doteq N \int_{B_N} \mid \nabla\phi(x,y) \mid \mathrm{d}x\,\mathrm{d}y \tag{3.7}$$

kde B_N je úzký pás definovaný

$$B_N \coloneqq \left\{ (x,y)\,; \mid \phi(x,y) \mid \leq \frac{1}{2N} \right\}$$

Střed tohoto pásu tvoří křivka $\partial\omega : \phi(x,y) = 0$, nebo pokud ji parametrizujeme délkou s

$$\partial\omega : \; s \mapsto \mathbf{z}(s) \quad (0 \leq s \leq L \coloneqq Length(\partial\omega))$$

Vezmeme bod $\mathbf{p} \coloneqq \mathbf{z}(s) \in \partial\omega$. Směr pásu je $\dot{\mathbf{z}}(s)$ a normálový směr je $\nu \coloneqq \nabla\phi(\mathbf{z}(s))$. Začneme v bodě $\mathbf{p}$ a půjdeme ve směru normály ν až dosáhneme hranice pásu B_N. Urazili jsem vzdálenost $h > 0$ takovou, že $\mid \nu \mid h = \frac{1}{2N}$. Z toho vyplývá, že šířka pásu $\rho(s)$ blízko $\mathbf{z}(s)$ je přibližně dána

$$\rho(s) = 2h = \frac{1}{N \mid \nu \mid} = \frac{1}{N \mid \nabla\phi(\mathbf{z}(s)) \mid}$$

Proto máme

$$N \int_{B_N} \mid \nabla\phi(x,y) \mid \mathrm{d}x\,\mathrm{d}y \doteq N \int_0^L \rho(s) \mid \nabla\phi(\mathbf{z}(s)) \mid \mathrm{d}s = \int_0^L \mathrm{d}s = Length(\partial\omega)$$

Po dosazení do vztahu (3.7) dostáváme požadovaný vztah.

Po dosazení vztahů (3.3) - (3.6) do funkcionálu $F(c_1, c_2, C)$ definovaného vztahem

(3.1) získáme formulaci vodorovných řezů funkcionálu $F(c_1, c_2, \phi)$ ve tvaru

$$F(c_1, c_2, \phi) = \mu \int_\Omega \delta(\phi(x,y)) \mid \nabla\phi(x,y) \mid dx\,dy + \nu \int_\Omega H(\phi(x,y))\,dx\,dy \tag{3.8}$$

$$+ \lambda_1 \int_\Omega \mid u(x,y) - c_1 \mid^2 H(\phi(x,y))\,dx\,dy$$

$$+ \lambda_2 \int_\Omega \mid u(x,y) - c_2 \mid^2 (1 - H(\phi(x,y)))\,dx\,dy \tag{3.9}$$

Zvolíme pevné ϕ a budeme hledat minimum funkcionálu $F(c_1, c_2, \phi)$ vzhledem k c_1 a c_2. Hodnotu c_1 dostaneme derivováním funkcionálu podle c_1, kdy všechny ostatní proměnné ve funkcionálu bereme jako parametry díky pevnému ϕ. Z výrazu

$$\frac{\partial F}{\partial c_1} = \lambda_1 \int_\Omega -2uH + 2c_1 H = 0$$

vyjádříme c_1. Stejným způsobem získáme vyjádření c_2 . Nyní je lze vyjádřit jako konstantní funkce proměnné ϕ ve tvaru

$$c_1(\phi) = \frac{\int_\Omega u(x,y)\, H(\phi(x,y))\,dx\,dy}{\int_\Omega H(\phi(x,y))\,dx\,dy} \tag{3.10}$$

pokud je $\int_\Omega H(\phi(x,y))\,dx\,dy > 0$ (tj. pokud křivka bud mít neprázdný vnitřek v Ω) a

$$c_2(\phi) = \frac{\int_\Omega u(x,y)\,(1 - H(\phi(x,y)))\,dx\,dy}{\int_\Omega (1 - H(\phi(x,y)))\,dx\,dy} \tag{3.11}$$

pokud je $\int_\Omega (1 - H(\phi(x,y)))\,dx\,dy > 0$ (tj. pokud křivka má neprázdný vnějšek v Ω). Jak je vidět, c_1 a c_2 jsou skutečně průměry hodnot v daných oblastech tj.:

$$c_1(\phi) = \text{průměr}(u) \text{ v } \{\phi \geq 0\}$$
$$c_2(\phi) = \text{průměr}(u) \text{ v } \{\phi < 0\}$$

K řešení vyvíjejících se křivek v čase t tak, aby minimalizovali funkcionál J využijeme gradientního toku. Zavedeme nyní funkci $\phi(t,x,y) = \phi_t(x,y)$, kde $t \geq 0$. $\phi(0,x,y)$ je počáteční křivka. Následující vývojová rovnice (evolution equation)

$$\frac{\partial \phi}{\partial t} = -\frac{\partial J}{\partial \phi} \tag{3.12}$$

je gradientní tok (viz[16]). Tedy křivka ϕ se bude vyvíjet v čase podle (3.12) tak, aby

se minimalizoval funkcionál (3.8). Vyjádříme

$$\frac{\partial J}{\partial \phi} = \delta(\phi)\left[-\mu \mathrm{div}\left(\frac{\nabla \phi}{\mid \nabla \phi \mid}\right) + \nu + \lambda_1 (u - c_1)^2 - \lambda_2 (u - c_2)^2\right]$$

Proces největšího poklesu pro minimalizaci funkcionálu J je následující gradientní tok:

$$\frac{\partial \phi}{\partial t} = \delta(\phi)\left[\mu \mathrm{div}\left(\frac{\nabla \phi}{\mid \nabla \phi \mid}\right) - \nu - \lambda_1 (u - c_1)^2 + \lambda_2 (u - c_2)^2\right]$$

Pro vývoj křivky C v normálovém směru platí

$$\begin{cases} \frac{\partial \phi}{\partial t} = \delta(\phi)\left[\mu \mathrm{div}\left(\frac{\nabla \phi}{|\nabla \phi|}\right) - \nu - \lambda_1 (u - c_1)^2 + \lambda_2 (u - c_2)^2\right] & t \in (0, \infty), (x, y) \in \Omega \\ \phi(0, x, y) = \phi_0(x, y) & (x, y) \in \Omega \end{cases} \tag{3.13}$$

Funkce ϕ, která minimalizuje funkcionál J, musí být i jeho stacionárním bodem, tedy musí platit

$$\frac{\partial J}{\partial \phi} = \delta(\phi)\left[-\mu \mathrm{div}\left(\frac{\nabla \phi}{\mid \nabla \phi \mid}\right) + \nu + \lambda_1 (u - c_1)^2 - \lambda_2 (u - c_2)^2\right] = 0 \tag{3.14}$$

.

3.1.3 Numerická aproximace modelu

Pro numerický výpočet musíme zavést funkce H_ε a δ_ε. H_ε je alespoň $C^2\left(\overline{\Omega}\right)$ regularizace Heavisidovy funkce H a δ_ε je regularizací Diracovy míry δ, pro kterou platí $\delta_\varepsilon = H'_\varepsilon$.

Vezmeme si konkrétní $C^\infty\left(\bar{\Omega}\right)$ regularizaci H ve tvaru

$$H_\varepsilon(z) = \frac{1}{2}\left(1 + \frac{2}{\pi}\arctan\left(\frac{z}{\varepsilon}\right)\right)$$

a regularizaci δ vezmeme δ_ε jako derivaci H_ε, tedy

$$\delta_\varepsilon(z) = \frac{1}{\pi}\frac{\varepsilon}{\varepsilon^2 + z^2}$$

Jiné regularizace nemají tak dobré vlastnosti a často hledají pouze lokální minima (viz [4])

K diskretizaci diferenciálních rovnic použijeme konečné implicitní diferenční formule a diskretizaci operátoru divergence uvedenou v [17]. Nechť h je prostorový krok, $\triangle t$ je

krok v časové proměnné. Uzly označme $(x_i,\, y_j) = (ih,\, jh)$, pro $1 \leq i,\, j \leq M$. Nechť $\phi^n_{i,j} = \phi\left(n\triangle t, x_i, y_j\right)$ je aproximace $\phi\left(t, x, y\right)$ pro $n \geq 0$, kde $\phi^0 = \phi_0$. Konečné diference mějme ve tvaru

$$\begin{aligned} \triangle^x_- \phi_{i,j} &= \phi_{i,j} - \phi_{i-1,j}, \qquad & \triangle^x_+ \phi_{i,j} &= \phi_{i+1,j} - \phi_{i,j} \\ \triangle^y_- \phi_{i,j} &= \phi_{i,j} - \phi_{i,j-1}, \qquad & \triangle^y_+ \phi_{i,j} &= \phi_{i,j+1} - \phi_{i,j} \end{aligned}$$

Diskretizace rovnice (3.13) vývoje křivky vyjádříme ve tvaru:

$$\begin{aligned} \frac{\phi^{n+1}_{i,j} - \phi^n_{i,j}}{\triangle t} \;=\; & \delta_h\left(\phi^n_{i,j}\right)\left[\frac{\mu}{h^2}\triangle^x_-\left(\frac{\triangle^x_+\phi^{n+1}_{i,j}}{\sqrt{\left(\triangle^x_+\phi^n_{i,j}\right)^2/\left(h^2\right) + \left(\phi^n_{i,j+1} - \phi^n_{i,j-1}\right)^2/\left(2h\right)^2}}\right)\right. \\ & + \frac{\mu}{h^2}\triangle^y_-\left(\frac{\triangle^y_+\phi^{n+1}_{i,j}}{\sqrt{\left(\triangle^y_+\phi^n_{i,j}\right)^2/\left(h^2\right) + \left(\phi^n_{i+1,j} - \phi^n_{i-1,j}\right)^2/\left(2h\right)^2}}\right) \\ & \left. -\nu - \lambda_1\left(u_{i,j} - c_1\left(\phi^n\right)\right)^2 + \lambda_2\left(u_{i,j} - c_2\left(\phi^n\right)\right)^2\right] \end{aligned} \tag{3.15}$$

Algoritmus:

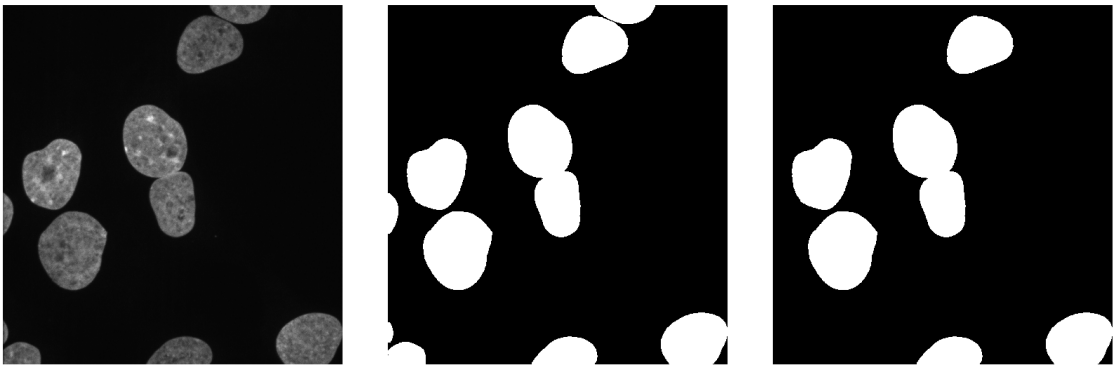
1. Zavedeme počáteční $\phi^0 = \phi_0$, $n = 0$
2. Vypočteme $c_1\left(\phi^n\right)$ a $c_2\left(\phi^n\right)$ podle (3.10) a (3.11)
3. Vypočteme ϕ^{n+1} z rovnice 3.15
4. Zkontrolujeme, zda je řešení stacionární (splňuje rovnici (3.14)), neboli pokud pravá strana rovnice 3.15 je menší než dostatečně malé $\varepsilon > 0$.
   Pokud řešení není stacionární vrátíme se ke kroku 2. a postup opakujeme.

4 Program využívající metodu aktivních kontur

4.1 Popis postupu

Nyní popíšeme postup programu pro detekci jader využívající metodu aktivních kontur.

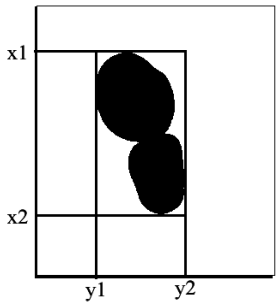
Postup je následující. Načteme snímek ve stupních šedé. V dalším kroku se provede prahování tohoto snímku. Před prahováním se takzvaně vyplní díry pomocí funkce `imfill` s parametrem `holes`. Eliminují se tak body uvnitř jader, které jsou tmavé a po prahování by vytvořily malé oblasti uvnitř objektů, které by byly černé. Tyto oblasti by byly přiřazovány k pozadí a při dalším postupu by způsobovaly problémy. Pokud bychom vyplnění provedli po prahování došlo by i k nežádoucímu vyplnění částí uvnitř uzavřených shluků. Poté již provedeme prahování a následně provedeme morfologickou operaci `open`. Tato operace kombinuje erozi a rozšíření oblastí a tím dochází k určitému vyhlazení hranic objektů.



(a) (b) (c)

Obrázek 4.1: Kroky algoritmu. (a) originální snímek ve formátu uint8, (b) snímek po prahování, (c) snímek po odstranění malých objektů

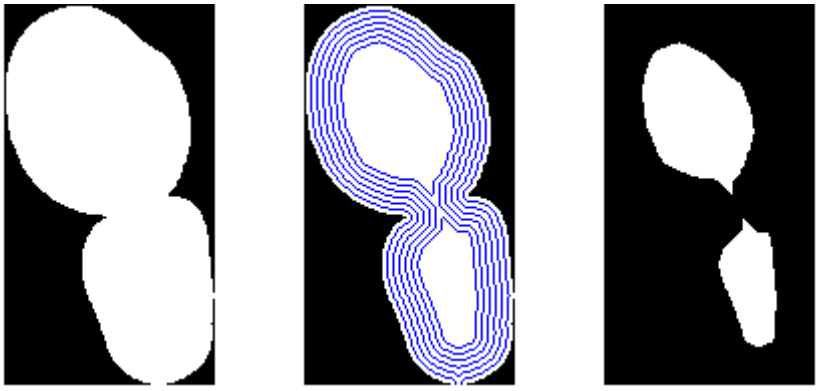
Poté následuje hlavní proces programu. Nadále budeme pracovat s binárním snímkem získaným z prahování. Odstraníme z něj příliš malé objekty. Zavedeme si obraz, do kterého budeme zapisovat nalezená jednotlivá jádra. Převedeme zde osamocená jádra na snímku a následně budeme rozdělovat shluky. Abychom nemuseli v každém kroku pracovat s celým snímkem, provedeme ořez pouze na jednotlivé



Obrázek 4.2: Hodnoty ořezu objektu

objekty. Pro každý objekt dostaneme polohu ohraničujícího obdélníku v celém snímku, tj. x1, x2, y1, y2.

Jelikož budeme používat metodu aktivních kontur, potřebujeme pro tuto metodu počáteční oblast. Neboli binární snímek, na kterém je bílá oblast uvnitř počáteční křivky, která je potřeba pro metodu aktivních kontur (viz kapitola 3). Pro počáteční oblasti blízko hranice objektu získáme lepší výsledky. Tuto oblast získáme postupnou erozí objektu a tu provádíme tak dlouho, dokud se oblast nerozdělí alespoň na dvě části (Obrázek 4.3. (b)) V dalším kroku se bude provádět metoda aktivních kontur postupně pro jednotlivé části, které jsme dostali v předchozím kroku erozí. Dostaneme tak třírozměrné pole, které má tolik vrstev kolik je oblastí po provedení eroze (minimálně dvě) a každá vrstva obsahuje výsledek po metodě aktivních kontur pro jednotlivé počáteční oblasti.



(a) (b) (c)

Obrázek 4.3: (a) snímek obřezaný na daný objekt, (b) posloupnost oblastí z imerod, (c) výsledné počáteční oblasti

Předchozím postupem dostaneme požadované oblasti, které se z části překrývají (Obrázek 4.4 (a)), pokud jsou sousedící. Nyní vezmeme všechny možné dvojice oblastí a uděláme jejich průnik. Pokud je neprázdný provedeme jeho průnik s hranicí objektu, a tak získáme dvě množiny bodů. Z těchto množin vybereme dva body, které mají mezi sebou minimální vzdálenost (Obrázek 4.4 (c)). Tyto body vezmeme jako body řezu a provedeme v nich lineární řez. Pokud došlo k řezu, vrátíme rozřezaná jádra zpět do binárního snímku a postup opakujeme pro další objekt.

(a) (b) (c)

Obrázek 4.4: (a) Metoda aktivních kontur. Zelené jsou počáteční oblasti, červeně a fialově jsou ohraničeny výstupní oblasti z metody aktivních kontur. (b) Průnik oblastí. Bílá oblast je průnik dvou oblastí po metodě aktivních kontur. (c) Body řezu. Bílá je hranice průniku, která je společná s hranicí původního objektu, červeně jsou vyznačeny body, kde se provede řez

4.2 Parametry

Na začátku algoritmu se zadává několik parametrů a některé parametry se objevují dále v algoritmu.

minarea Tento parametr označuje minimální počet pixelů, který by měly mít jednotlivá jádra nebo objekty určené k dělení. Všechny objekty, které mají počet pixelů menší se z černobílého obrázku bw smažou, jde převážně o nečistoty a necelá jádra. Tento parametr závisí na daných snímcích.

solidita Soliditu vypočítáme ze vzorce

$$\text{solidita} = \frac{\text{počet pixelů objektu}}{\text{počet pixelů konvexního obalu objektu}}$$

Její hodnota je v rozmezí $[0, 1]$, konvexní objekty mají soliditu rovnu 1. Jádra mají podobný tvar s velkou soliditou, objekty s menší soliditou se pokusíme rozdělit. Soliditu jsem nastavila na 0.95 jelikož jádra mají poměrně kulatý tvar. S takto vysokou soliditou můžeme ztratit některá jednotlivá jádra, která nemají požadovanou soliditu. Pokud ji ale snížíme například na 0.9, získáme tak více jednotlivých jader, ale některé dvojice budou považovány za jednotlivá jádra a nebudou rozděleny.

maxarea Tento parametr určuje minimální počet pixelů pro objekty na dělení. Pokud

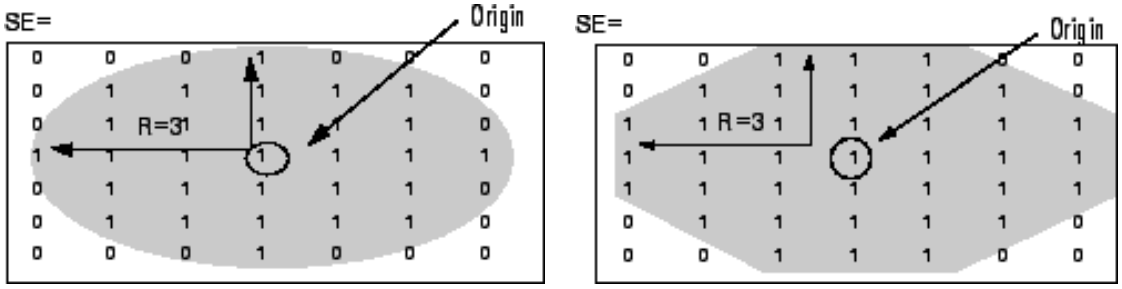
je objekt málo solidní, ale má počet pixelů menší než daný parametr, často se nejedná o objekt určený k dělení, a proto jej smažeme. Tento parametr stejně jako minarea záleží na daných snímcích, můžeme ho zvolit například jako dvojnásobek minarea.

hladkost Parametr hladkosti se vztahuje k metodě aktivních kontur. Více o tomto parametru je uvedeno u popisu funkce activecontour v 4.6 na straně 33 a její případné volby v části 5.2 na straně 39.

struktura pro imerode a imdilate V průběhu algoritmu se používají několikrát funkce `imerode` a `imdilate`. Funkce `imerode`, která je v Matlabu, vrací obraz s objektem získaným z objektu z původního snímku, pro který v každém pixelu na hranici odmažeme okolí. Okolí, které se má odmazávat, je dáno nějakou strukturou. Podobně funguje metoda `imdilate`, kde naopak pro každý pixel na hranici přidáváme okolí, které zadáváme nějakou strukturou. Pro zadávání této struktury slouží v Matlabu funkce `strel`, která se volá

$$\texttt{se} = \texttt{strel}(\acute{}\texttt{tvar}\acute{},\texttt{parametr})$$

V programu jsem nejčastěji používala strukturu okolí s tvarem disk a octagon. Pro tvar disk je parametr poloměr R disku. Pro octagon je parametr R, který určuje vzdálenost od originálního bodu ke stěnám osmistěnu. Parametr R u tvaru octagon musí být dělitelný třemi. Více o této funkci pro vytvoření okolí naleznete v [18].



(a) (b)

Obrázek 4.5: Struktury pro zadání okolí: (a) okolí disk a poloměrem R=3, (b) okolí octagon s parametrem R=3. (zdroj [18])

Konkrétní volby těchto struktur lze zvolit různé dle potřeby a charakteru analyzovaných obrazů. Svoje volby jsem volila buď odhadem nebo odzkoušením na konkrétních datech. V algoritmu jsem je napsala do závorek.

4.3 Algoritmus

Následující algoritmus popisuje program, který je přiložený na CD ve složce - program s active contour. Spouštění se provádí skriptem `spousteni.m`, kde se zadává název snímku a parametry. Jako vstup zadáme název snímku a získáme výstup binární snímek `jádro` s jednotlivými rozdělenými jádry.

1. Příprava, inicializace

   (a) zadáme parametry

   `minarea` = minimální počet pixelů jednotlivých jader

   `solidita` = minimální solidita jednotlivých jader

   `maxarea` = minimální počet pixelů u kandidátů na dělení

   `hladkost` = parametr hladkosti pro metodu aktivních kontur

   (b) načteme obrázek ve stupních šedé a označme ho `image`

   (c) prahování

   vyplníme díry v `image` pomocí `imfill`

   provedeme prahování `image` a výsledný binárním snímek označme `bw`

   vyhladíme `bw` pomocí `bwmorph(open)`

   (d) Zavedeme černobílý obrázek `jádro` pro ukládání jednotlivých nalezených jader

2. **while** (počet objektů v `bw` > 0)

   odstraníme objekty z `bw` s menším počtem pixelů než je `minarea`

   zjistíme počet objektů `num` v `bw` a jejich rozmístění pomocí `bwlabel`

   zjistíme velikost `velikost`(i) a soliditu `sol`(i) pro $i-$ tý objekt

   for i = 1 **to** `num`

   smažeme i-tý objekt z `bw`

   if (`sol`(i) $>$ `solidita`)

   if (průnik objektu s hranicí obrazu je nulový)

   zapíše se i-tý objekt do matice `jádro`

   elseif (`velikost`(i) $>$ `maxarea`)

provedeme **proces rozdělení.** Pokud dojde k rozdělení, rozdělený objekt se vrátí do `bw`.

end if

end if

end for

end while

4.4 Proces rozdělení

1. Provedeme ořez pouze na daný objekt

   zjistíme pozici ohraničujícího obdélníku objektu `x1, x2, y1, y2` v `bw`

   zavedeme obraz `bwi = bw(x1:x2,y1:y2)`

2. Počáteční oblast

   zavedeme struktura `se` pro `imerode` (octagon,3)

   `počáteční:=bwi`

   while (počet objektů v `počáteční` $\leq$ počet objektů v `bwi`) & ...

   & (počet objektů v `bwi` $\neq 0$)

   provádíme `imerode` pro `počáteční` se strukturou `se`

   end while

3. **if** (počet kroků v předchozím while-cyklu ==1)

   dělení provedeme pomocí rozšíření oblastí (více v části 4.5 na straně 33)

   else

4. Metoda aktivních kontur

   for i=1 : počet objektů v `počáteční`

   provedeme metodu aktivních kontur Chan-Vese na obrazu `bwi` pro i-tý objekt z `počáteční` a s parametrem `hladkost`. Výsledek zapíšeme do i-té vrstvy pole `snake`.

   end for

5. Samotné rozřezání objektů

for (všechny dvojice vrstev ve `snake`)

průnikem dvojce oblastí ze `snake` získáme výslednou oblast `průnik`

if (`průnik` je neprázdný)

nalezneme hranici průniku

rozšíříme hranici průniku pomocí `imdilate` (disk,5)

provedeme průnik rozšířené hranice průniku s hranicí původního objektu v `bwi` a označme ho `hraniční`

if (`hraniční` je neprázdný)

if (počet objektů v `hraniční` (8-okolí) $>$ 2)

while (počet objektů v `hraniční` (8-okolí) $>$ 2)

smažeme objekt s nejmenší velikostí

end while

elseif (počet objektů v `hraniční` (8-okolí) $<$ 2)

nejsou body pro dělení a vektor `body` nastavíme na prázdný

end if

nalezneme dva body v `hraniční` z jednotlivých objektů s minimální vzdáleností a vektor `body` obsahuje jejich polohu

if (vzdálenost bodů je příliš malá např. 3)

nejsou vhodné body pro dělení a vektor `body` nastavíme na prázdný

end if

if (vzdálenost bodů je příliš velký (např. 2/3 délky vedlejší poloosy elipsy, která má stejný normalizovaný druhý centrální moment jako oblast))

nejsou vhodné body pro dělení, vektor `body` nastavíme na prázdný

end if

if (`body` je neprázdný)

provedeme lineární řez v `bwi` mezi body z vektoru `body`

`dělení` = 1

end if

end if

end if

end for

if (`dělní` = 1)

`bwi` s rozdělenými objekty zapíšeme zpět do `bw`

end if

4.5 Rozdělení pomocí rozšíření oblasti

Tento postup se používá v algoritmu, pokud se počáteční oblast získá hned po prvním kroku imerode. To znamená, že jádra jsou spojeny pouze několika málo pixely a použitím tohoto postupu se vyhneme použití metody aktivních kontur, která je více časově náročná.

Označme si oblast, se kterou budeme pracovat jako `počáteční`. Vždy to bude ta oblast, která má větší soliditu. Pomocí funkce imdilate vytvoříme dvě další oblasti.

`minus` = imdilate `počáteční` se strukturou `seminus` (disk,5)

`plus` = (imdilate `počáteční` se strukturou `seplus` (disk,4)) průnik s (`bwi`)

Poté od `bwi` odečteme `minus` a k tomuto přičteme `plus`. Dostaneme tedy řez o šíři rozdílu velikosti struktur jak můžeme vidět na Obrázku 4.6.

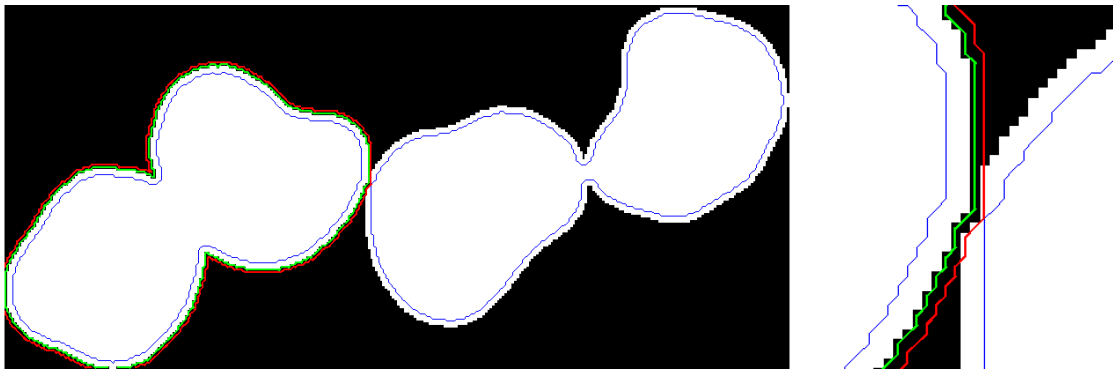
4.6 Analýza obrazu v Matlabu

Nyní si popíšeme několik funkcí implementovaných v Matlabu, které jsou důležité pro analýzu obrazu a které používám v programu.

Metoda aktivních kontur

Syntaxe

`bw` = `activecontour` (`A`, `mask`, `method`, `SmoothFactor`)



(a) (b)

Obrázek 4.6: (a) Objekty rozdělené po jednom kroku imerode. Modré jsou hranice oblastí po jednom kroku imerode, červená je hranice oblasti `minus`, zelená je hranice oblasti `plus`, (b) stejná situace v detailu.

Vstup:

A - originální obrázek ve stupních šedé určen k segmentaci

mask - binární obrázek s počáteční oblastí stejné velikosti jako originální obrázek A. Objekt je bílý a pozadí černé. Pokud mask obsahuje oblasti s dírami výsledky mohou být nepředvídatelné, a tak je lepší použít příkaz imfill. Pro lepší výsledky by počáteční oblasti měly být u metody edge vně objektu a u metody Chan-Vese uvnitř objektu.

method - typ metody aktivních kontur použité k segmentaci. Zadává se jako řetězec 'Chan-Vese' nebo 'edge'. Výchozí je nastavena metoda 'Chan-Vese'. Metoda Chan-Vese nemusí vždy nalézt všechny objekty. Pokud jsou na obrázku dva objekty, jeden světlejší a druhý tmavší než pozadí, metoda může nalézt pouze jeden objekt.

SmoothFactor - Stupeň hladkosti hranic segmentovaných oblastí. Je to kladná konstanta. Vyšší hodnoty zaručují větší hladkost hranice segmentovaného objektu, ale může vyhladit jemné detaily. Nižší hodnoty vytváří méně hladké oblasti ale mohou zachytit jemnější detaily. Defoltní hodnota závisí na typu metody, tj. 0 pro metodu Chan-Vese a 1 pro metodu edge.

Výstup:

bw - binární obrázek stejné velikosti jako vstupní obrázek A. Segmentovaný objekt je bílý (true) a pozadí je černé (false)

Jednoduché prahování - volba prahu

Globální prahování používající Otsuovu metodu

Syntaxe

$$\texttt{level} = \texttt{graythresh}\,(\texttt{I})$$

Vstup:

I - snímek ve stupních šedé, na kterém chceme provést prahování

Výstup:

level - normovaná hodnota prahu typu double v rozmezí [0; 1], kterou můžeme použít k převodu na binární snímek pomocí im2bw.

Prahování - převod na binární snímek

Syntaxe

$$\texttt{BW} = \texttt{im2bw}\,(\texttt{I},\, \texttt{level})$$

Vstup:

I - vstupní obraz ve stupních šedé nebo RGB obraz. Pokud je vstupní snímek ve formátu RGB, snímek se nejprve převede na snímek ve stupních šedé a následně se provede prahování.

level -hodnota prahu v rozmezí [0; 1], kterou můžeme získat funkcí graythresh. Pokud hodnotu nezadáme je nastavena defoltní 0.5.

Výstup:

BW - výstupní binární snímek po prahování stejné velikosti jako vstupní snímek.

Víceprahové prahování - volba prahu

Víceprahové prahování používající Otsuovu metodu

Syntaxe

$$\texttt{thresh} = \texttt{multithresh}\,(\texttt{A})$$

$$\texttt{thresh} = \texttt{multithresh}\,(\texttt{A}, \texttt{N})$$

Vstup:

A - Vstupní snímek ve stupních šedé nebo RGB snímek. Pro RGB snímky je hodnota počítána z kombinací tří vrstev.

N - počet hodnot prahů. Doporučují se menší hodnoty $N < 10$. Maximální povolená hodnota je 20. Defoltní je 1.

Výstup:

thresh - výstupní $1 \times N$ vektor obsahující hodnoty prahů ve stejném typu jako vstupní snímek A

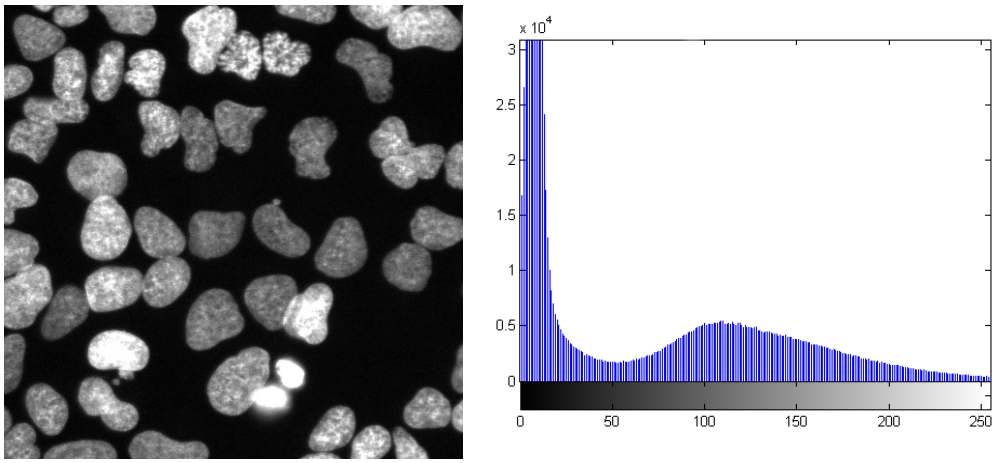
5 Výsledky

Program jsem testovala na konkrétních snímcích získaných z Laboratoře integrity genomu Ústavu molekulární a translační medicíny Lékařské fakulty Univerzity Palackého v Olomouci získaných v rámci výzkumu integrity genomu. Jsou snímány buňky kostního osteosarkomu z holenní kosti 15leté dívky s mírně diferencovaným sarkomem. Vzorek je obarven DAPI, což je fluorescenční barvivo, které se váže na DNA. Vzorek je snímám se 400 násobným zvětšením. Na snímcích jsou vidět pouze obarvená jádra buněk. Ze snímků lze vyčíst morfologie jader, jako je solidita, velikost, struktura a další. Můžeme je použít pro lokalizování jader při zkoumání integrity genomu jak je popsáno v úvodu. Lokalizování jednotlivých jader a rozdělení shluků je důležité, jelikož by byly výsledky nadhodnocené a zkreslené. Například by se mohlo zdát, že na snímku je méně jader a mají větší velikost a menší soliditu.

Tyto snímky jsou uloženy na přiloženém CD ve složce Testovací snímky.

5.1 Prahování

Jak jsem již uvedla v sekci 2, dosti důležité je správné nalezení objektů, tedy správné prahování. Zkoušela jsem několik metod pro automatické nalezení optimálního prahu. Všechny získané snímky mají podobný histogram. Každá z metod byla jinak účinná na daný typ histogramu.



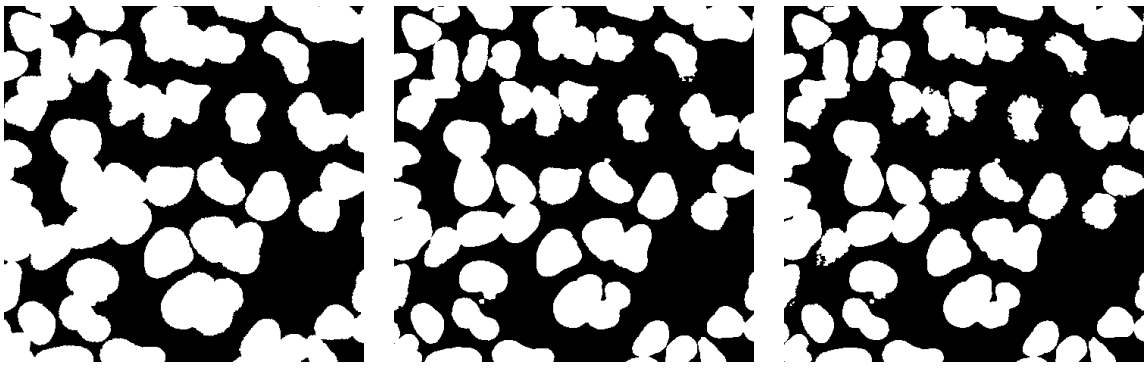
(a) (b)

Obrázek 5.1: (a) originální snímek ve formátu uint8 (b) histogram originálního snímku

Pokud za hodnotu prahu zvolíme medián hodnot histogramu, získáme příliš nízkou hodnotu, jelikož histogramy jsou nerovnoměrné a na počátku má histogram příliš vysoké

hodnoty. Dobré výsledky získáme pokud za hodnotu prahu vezmeme střední hodnotu hodnot histogramu.

Použitím základní iterativní metody uvedené v části 2.1 dostáváme obecně vyšší hodnoty.



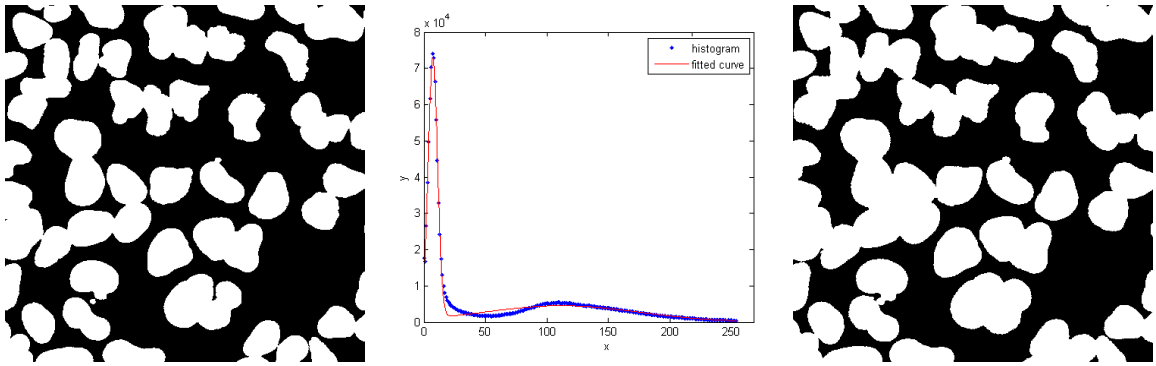
(a) (b) (c)

Obrázek 5.2: binární snímky po prahování s hodnotou prahu (a) rovnou mediánu ($t_{med} = 17$), (b) rovnou střední hodnotě ($t_{\mu} = 62$), (c) získanou iterační metodou ($t_{ad} = 78$)

Metoda prohlubní dává také poměrně dobré výsledky. Největší problém je v nalezení druhého maxima a tedy rozhraní, kde se má minimální hodnota hledat. Proto může být výhodnější proložit histogram křivkou. Daným histogramům nejlépe vyhovuje křivka, která je kombinací dvou Gaussových křivek a na této křivce budeme hledat minimum mezi dvěma vrcholy. Takto ale dostaneme příliš nízkou hodnotu prahu, je to způsobeno vysokými hodnotami na začátku histogramu, které se prokládají Gaussovým rozdělením s malým rozptylem a to má za příčinu příliš prudký pokles (Obrázek 5.3 (b)).

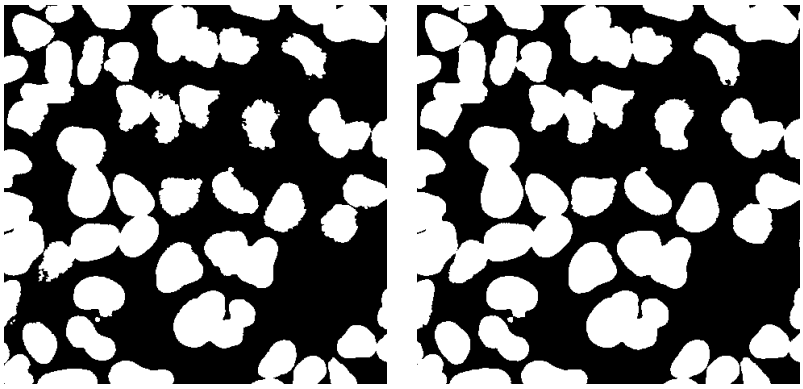
Klasická Otsuova metoda, která se v matlabu zadává `level = graythresh(I)`, dává vysoké hodnoty. Odzkoušením jsem v programu použila víceprahovou Otsuovu metodu pro dva prahy a hodnotu prahu zvolíme tu nižší.Tato metoda v Matlabu se volá příkazem `thresh = multithresh(A,N)`, kde `A` je originální obrázek, `N` značí počet prahů, tedy v našem případě 2. Výstup je dvouprvkový vektor, ze kterého bereme první prvek.

Víceprahová Otsuova metoda a metoda, kde za hodnotu prahu bereme střední hodnotu, dávají ze všech metod nejlepší výsledky. Pro některé snímky jsem dostávala lepší výsledky pro práh roven střední hodnotě, ale obecně lepší výsledky jsem získala po použití víceprahové Otsuové metodě.



(a) (b) (c)

Obrázek 5.3: (a) binární snímek po prahování s prahem získaným jako minimum histogramu mezi vrcholy ($t_{min} = 55$) (b) proložení histogramu křivkou (kombinace dvou Gaussových křivek) (c) binární obraz po prahování s prahem získaným jako minimum mezi dvěma vrcholy křivky proložené histogramem ($t_{min\_Gauss} = 22$)



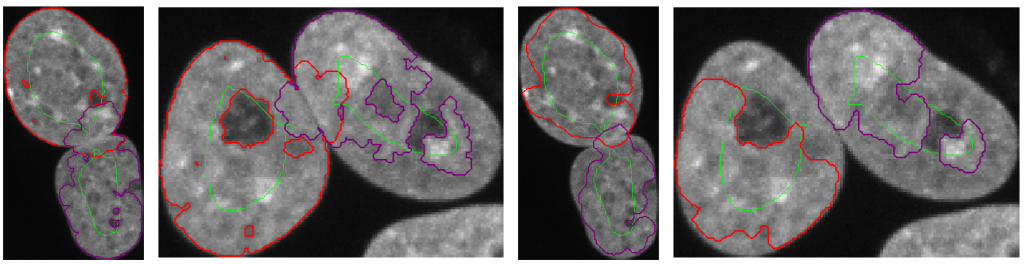
(a) (b)

Obrázek 5.4: binární snímky po prahování s prahy získanými (a) klasickou Otsuovou metodu ($t_{Ostu1} = 77$), (b) víceprahovou Otsuovou metodou ($t_{Otsu2} = 60$)

5.2 Metoda aktivních kontur

Další důležitou částí algoritmu je získání oblastí metodou aktivních kontur. Pro tento krok jsem využila funkci acivecontour, která je implementovaná v Matlabu, s druhem metody Chan-Vese. Nejdříve jsem tuto metodu aplikovala na originální snímek ve stupních šedé ve formátu uint8. Výstupní oblasti pro tyto vstupní snímky byly vzdálené od skutečných hranic objektů kvůli struktuře objektů na snímcích. Pokud jsem zvolila hodnotu parametru hladkosti vysokou, tak se výsledné oblasti často ani nepřekrývaly, ale neměly ani dostatečně hladký tvar. Pro nízké hodnoty parametru hladkosti, například hodnoty okolo 0.2 , jsem dostala překrývající se oblasti, ale nebyly často dostatečné.

Abych vyřešila problém se strukturou objektů, tak jsem místo originálních snímků



(a) (b) (c) (d)

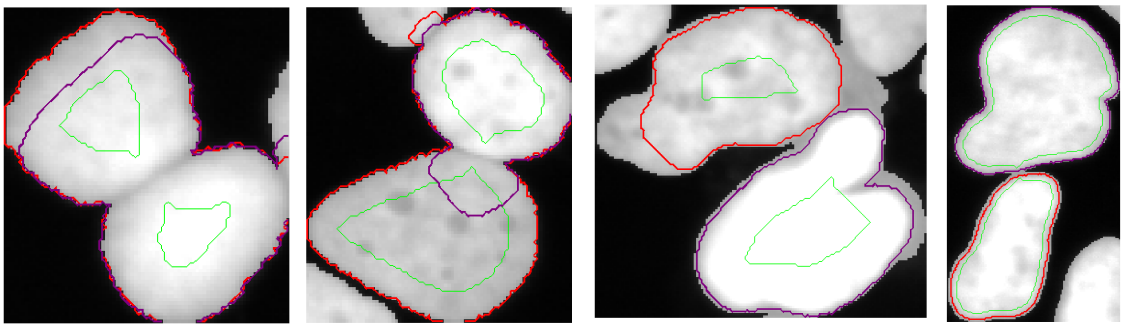
Obrázek 5.5: Metoda aktivních kontur, kde jako vstupní obrazy jsou originální snímky ve stupni šedé: (a) a (b) s parametrem hladkosti 0.3, (c) a (d) s parametrem hladkosti 0.7

zkusila vzít jako vstupní obrazy snímky, které byly konvexní kombinací originálního snímku a binárního snímku po prahování.

$$\text{image} = \alpha \cdot \text{bw} + (1 - \alpha) \cdot \text{original}$$

Metoda s tímto vstupním obrazem dává nejlepší výsledky při nižším parametru hladkosti okolo 0.3. Bohužel i zde se občas vyskytovaly problémy. Například pokud jeden objekt ze shluku byl výrazně světlejší nebo tmavší, tak po kombinaci tento rozdíl světelnosti přetrvává. Může se stát, že v takovém případě nebudeme mít požadovaný průnik oblastí pro nalezení bodů řezu, jak můžeme vidět na Obrázku 5.6 (c) a (d). Pokud bude alespoň jedna oblast dostatečně hladká a blízko hranice může se tato situace zachránit rozdělením rozšířením oblastí jak je uvedeno v části 4.5. Častější problém při nízkém parametru hladkosti bývá, že u některých objektů jsou výsledné oblasti po metodě aktivních kontur příliš velké a výsledný průnik může být i celý shluk jak je vidět na Obrázku 5.6 (a) a (b). V takovém případě nelze nalézt body pro dělení. Kdybychom zvýšili parametr hladkosti na 0.6, problém s příliš velkými výslednými oblastmi zmizí, ale naroste problém, kde výsledné oblasti nebudou mít společný průnik.

Nakonec jako nejspolehlivější se ukázalo použít binární vstupní obrazy , které získám po prahování. Nyní musíme oproti předchozím případům nastavit parametr hladkosti na vyšší hodnotu, a to okolo 0.6, jelikož u binárních obrazů má metoda větší tendenci stahovat se k hranicím objektu. Při nízkých hodnotách bychom často dostávali jako výsledné oblasti opět znovu celé shluky.



(a) (b) (c) (d)

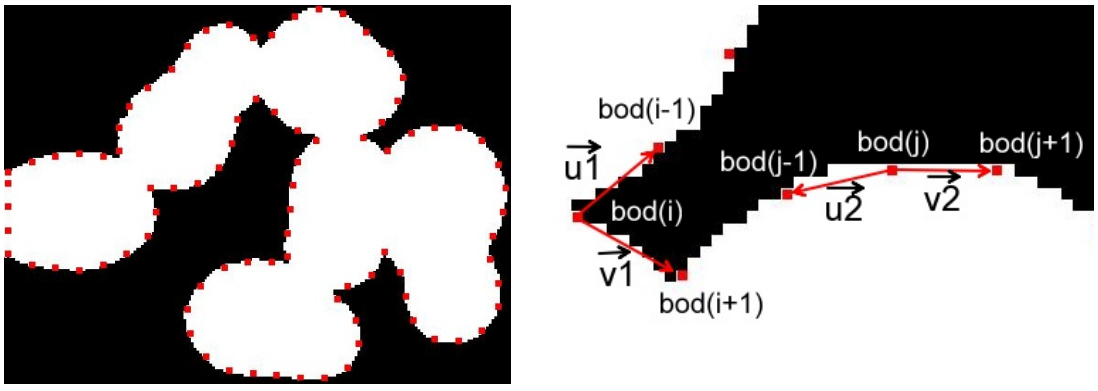
Obrázek 5.6: Snímky po metodě aktivních kontur na snímcích kombinací binárních a originálních snímků s parametrem $\alpha = 0.5$, zelené jsou počáteční oblasti a fialově a červeně jsou ohraničené výsledné oblasti. (a) a (b) metoda s parametrem hladkosti 0.3, kde výsledné oblasti jsou příliš velké. (c) a (d) metoda s parametrem 0.6, kde výsledné oblasti nemají společný průnik.

6 Srovnání

V této části se zaměřím na porovnání programu, který používá metodu aktivních kontur a programu, který prohledává hranice objektů. Druhý program jsme vytvořili společně s Tomášem Fürstem. V předchozích částech jsem popsala jak funguje program založený na metodě aktivních kontur. Nyní popíši princip druhého programu a následně ukáži jejich nedostatky a srovnání výsledků na sadě testovacích snímků.

6.1 Popis programu založeného na prohledávání hranic objektů

Dělení shluků jader se provádí následujícím postupem. Hranice objektu se diskretizuje s určitým krokem, to znamená, že se bere jen každý například desátý bod hranice, jako na Obrázku 6.1. V každém i-tém bodě se vypočítá vektorový součin vektorů $\overrightarrow{u} = bod\,(i-1) - bod\,(i)$ a $\overrightarrow{v} = bod\,(i+1) - bod\,(i)$ a bude nás zajímat jeho třetí složka. Vektorový součin bude kolmý k oběma vektorům a bude mít první a duhou složku nulovou. Pokud bude úhel mezi vektory (měřený vně objektu) větší než 180°, vektor bude směřovat vzhůru a jeho třetí složka bude kladná. Pokud úhel mezi $\overrightarrow{u}$ a $\overrightarrow{v}$ bude menší než 180°, vektor bude směřovat dolů a jeho třetí složka bude záporná. Prohlubně jsou tedy zpravidla body, které mají třetí složku vektorového součinu co nejvíce zápornou. Dále v každém bodě vypočteme normovaný směr $\overrightarrow{u} + \overrightarrow{v}/\|\overrightarrow{u} + \overrightarrow{v}\|$, to je směr osy úhlu. Hodnota třetí složky vektorového součinu slouží k určení vhodných bodů k řezu a směr osy úhlu nám poslouží ke správnému spárování bodů pro řez.



(a) (b)

Obrázek 6.1: Diskretizovaná hranice. (a) červené body jsou body diskretizované hranice. (b) detail obrázku (a), kde jsou vyznačeny vektory pro dané body. V bodě bod(i) bude hodnota třetí složky vektorového součinu záporná. V bodě bod(j) bude hodnota třetí složky vektorového součinu kladná.

Zavede se nový vektor jehož velikost je rovna počtu bodů diskretizované hranice.Tento vektor bude sloužit jako indikátor pro body k dělení, kde 1 bude znamenat, že bod je vhodný bod dělení, v opačném případě bude hodnota 0. Nyní se bude každý bod srovnávat s jeho okolními body. Pokud hodnota třetí složky vektorového součinu v i-tém bodě je menší nebo rovna hodnotě třetí složce vektorového součinu v bodech $bod\,(i-1)$ i $bod\,(i+1)$, tak jej označíme 1. Dále se odstraní body, které mají hodnotu třetí složky vektorového součinu větší než daný limit. V dalším kroku se souvislé úseky bodů k dělení nahradí pouze jediným bodem. Pokud je v řadě vedle sebe několik jedniček bere se pouze prostřední a zbylé se smažou (nastaví se na 0). Poté, aby nebyly dělící body příliš blízko sebe, se procházejí zbylé body v pořadí od nejmenší (nejvíce záporné) hodnoty třetí složky vektorového součinu a v jejich okolí se ostatní body odstraní.

Tímto postupem jsme určily body, ve kterých se provede řez a nyní se budou správně párovat. Pro všechny kombinace těchto bodů se vypočítá jejich vzájemná vzdálenost a také to jak jsou vzájemně protilehlé, a to ze směrů os úhlů podle vztahu $\|směr\,(i) + směr\,(j)\,/2\|$. Hodnota této protilehlosti, může být z [0; 1] a čím menší je hodnota, tím jsou body více protilehlé. Dále ověříme zda mají dvojice bodů mezi sebou minimální vzdálenost vzhledem k bodům v jejich okolí. To znamená, že jejich vzdálenost musí být menší než vzdálenosti dvojic bodů $bod\,(i)$ a $bod\,(j-1)$, $bod\,(i)$ a $bod\,(j+1)$, $bod\,(i-1)$ a $bod\,(j)$, $bod\,(i+1)$ a $bod\,(j)$. Pokud tedy dvojice splňuje poslední podmínku a má dostatečně malou vzájemnou vzdálenost a jsou dostatečně protilehlé, provede se mezi těmito body lineární řez.

6.2 Nevýhody jednotlivých postupů

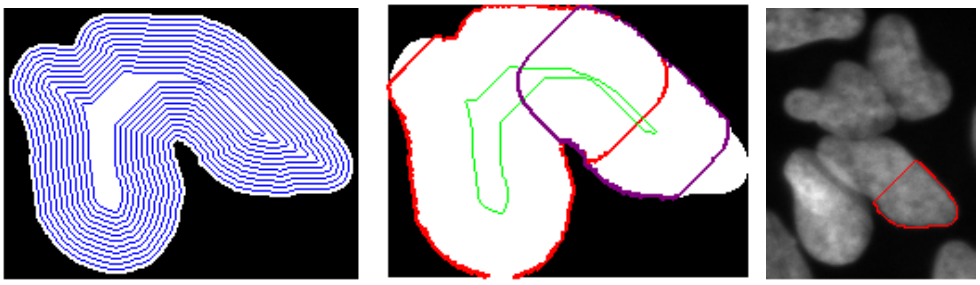
Program s metodou aktivních kontur

Hlavní nevýhodou tohoto programu je velká časová náročnost, která je způsobena časovou náročností funkce activecontour. Průběh jednoho volání funkce activecontour na mém počítači trval okolo 3 sekund. Tato funkce se navíc spouští jednotlivě pro každé jednotlivé jádro, které se oddělí ze shluku.

Další nevýhody vyplývají ze získávání počáteční oblasti pro metodu aktivních kontur. Jedna z nich je u úzkých dlouhých jader, které jsou spojeny s dalšími jádry delší stranou a při získávání počátečních oblastí funkcí imerode se odstraní jádro ještě před tím, než se rozdělí na dvě části jako na Obrázku 6.2. Nedojde tedy k řezu.Horší případ nastává také u úzkých podélných jader, kdy se erozí získají dvě oblasti tak, že výsledný řez prochází jádrem jako na Obrázku 6.3. Dochází tak k nechtěnému řezu.

(a) (b) (c)

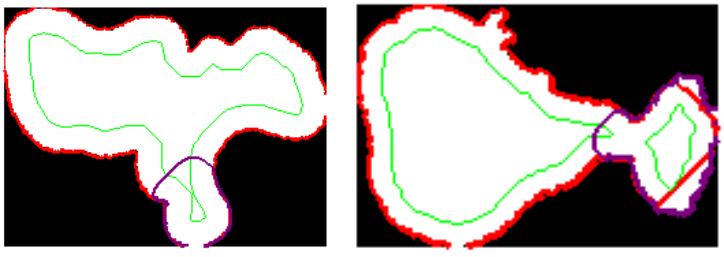
Obrázek 6.2: Nevýhoda odstranění jádra před rozdělením. (a) posloupnost výsledných oblastí po imerode, (b) metoda aktivních kontur pro získané počáteční oblasti označené zeleně, (c) posloupnost po imerode při druhém dělení, pouze jedna výstupní oblast, metoda aktivních kontur se neprovede, jelikož nemá počáteční oblasti.



(a) (b) (c)

Obrázek 6.3: Případ špatného řezu: (a) posloupnost výsledných oblastí po imerode, (b) metoda aktivních kontur pro získané počáteční oblasti označené zeleně, (c) výsledek po řezu

Jak jsem uvedla již v předchozí části, může se stát, hlavně u nízké hodnoty parametru hladkosti pro metodu aktivních kontur, že výsledné oblasti po této metodě jsou příliš velké jako na Obrázku 6.4. Tato situace se může občas objevit i pokud máme parametr hladkosti vysoký, jelikož výsledné oblasti dosti závisí na počátečních oblastech. Případů, ve kterých se vyskytují předchozí chyby, naštěstí není mnoho.



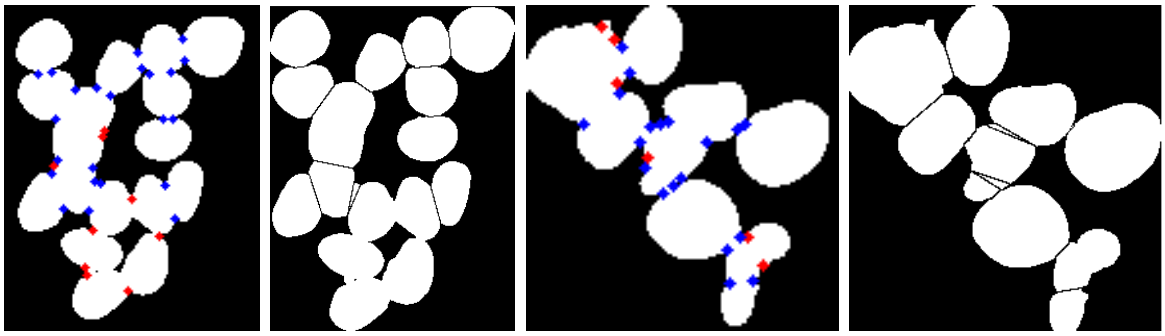
Obrázek 6.4: Příliš velké výsledné oblasti metody aktivních kontur

Tento program odděluje jádra postupně ze shluků oproti programu s prohledáváním hranic objektů, který dělení provádí pro celý shluk. To je nevýhoda v podobě zvýšení

časové náročnosti, ale díky tomu získáme často více rozdělených jader.

Program s prohledáváním hranice

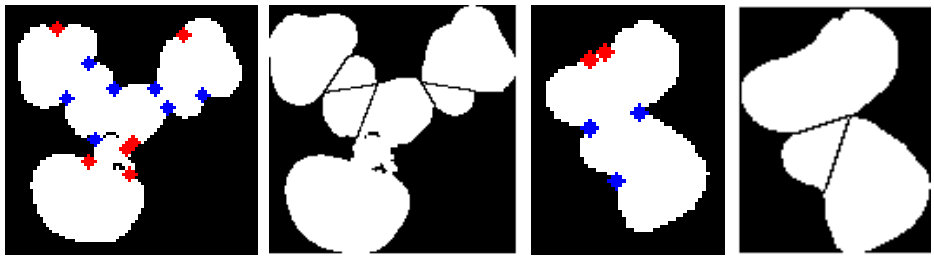
Jedna nevýhoda je, že se prochází pouze vnitřní hranice objektu. Pokud jádra tvoří uzavřený shluk nedojde k žádnému dělení, protože budou chybět body k dělení uvnitř tohoto shluku. Příklad můžeme vidět na Obrázku 6.5 (a) a (b).



(a) (b) (c) (d)

Obrázek 6.5: Chyba při uzavřených shlucích: (a) modré jsou body ve kterých proběhne řez, červené jsou body, které byly dalšími kandidáty na body řezu. (b) objekt po řezu. Případ více řezů z jednoho bodu: (c) modré jsou body ve kterých proběhne řez, červené jsou body, které byly dalšími kandidáty na body řezu, (d) objekt po řezu

Další nevýhody jsou způsobeny tím, že z jednoho bodu může vést více řezů a body pro řez jsou relativně blízko sebe. Jednou nevýhodou je, že mezi buňkami vznikne malý klínek nebo pruh, jak je vidět na Obrázku 6.5 (c) a (d). Tyto části se podmínkami na jednotlivá jádra eliminují a dostaneme řez mezi jádry, který je široký. V horším případě vede řez přes větší část jádra a dochází tak k nechtěným řezům jako na Obrázku 6.6.

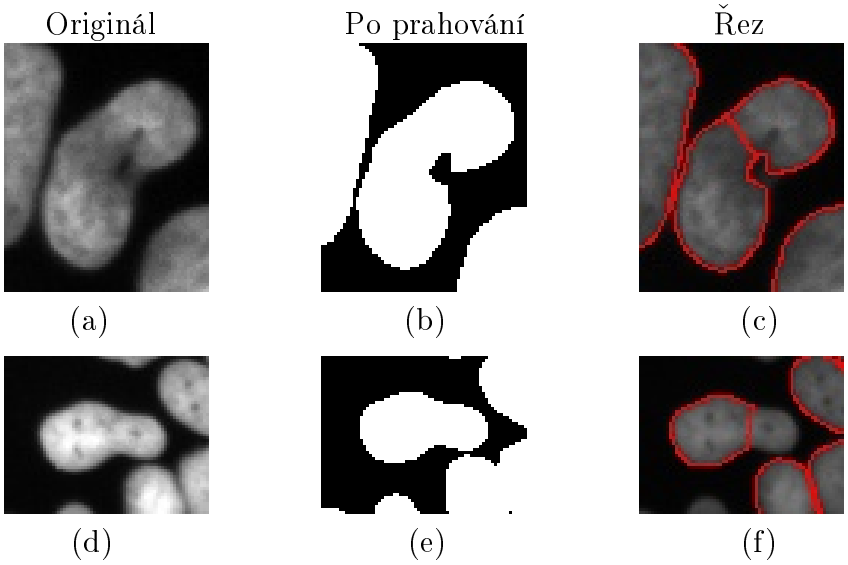


(a) (b) (c) (d)

Obrázek 6.6: Nechtěné řezy: (a) a (c) modré jsou body řezu, červené jsou kandidáti na řez, které nevyhověly podmínkám, (b) a (d) výsledný řez.

6.3 Srovnání výsledků

Nyní porovnám výsledky obou programů. Programy jsem testovala na vybraných snímcích z Laboratoře integrity genomu, které jsou uloženy na přiloženém CD ve složce Testovací snímky. Výsledné snímky jsou uloženy na CD ve složkách výsledky active contour a výsledky prohledávání hranic. Výsledky jsem zapsala do následující Tabulky 1. Časy trvání jsou jen orientační a pro srovnání programů. Počet chyb označuje pouze počet špatných nežádoucích řezů. Neberou se v úvahu chyby způsobeny nastavením hodnoty parametru solidity. To znamená, že shluk se nerozdělí, jelikož má vysokou soliditu. V závorce uvedu počet chyb, které jsou patrné až v porovnání s originálními snímky ve stupních šedé. Jedná se o špatné rozdělení způsobeno nedokonalostí prahování, jako v případě na Obrázku 6.7 (a), (b), (c), nebo špatné řezy způsobeny tvarem objektu. Jsou to především jádra protáhlá a jejich tvar na binárním snímku je podobný dvojici jader, jako na Obrázku 6.7 (d), (e), (f).



Obrázek 6.7: Příklady špatných řezů z testovacích snímků způsobených nedokonalostí prahování a tvarem jádra.

Dané výsledky jsem získala programy s uvedenými parametry

Pro snímky 1 - 20

Program s prohledáváním hranic :

```
pars.minarea = 2500
pars.solidity = 0.95
```

Program s metodou aktivních kontur

```
minarea = 2500
```

```
solidita = 0.95
maxarea = 6000
hladkost = 0.6
```

Pro snímky 21 a 22, které jsou jiné velikosti než předchozí snímky

Program s prohledáváním hranic:

```
pars.minarea = 1200
pars.solidity = 0.95
```

Program s metodou aktivních metod:

```
minarea = 1200
solidita = 0.95
maxarea = 3000
hladkost = 0.6
```

Srovnávala jsem dva programy na testovací sadě snímků. První program využívá k nalezení bodů k řezu metodu aktivních kontur a druhý prochází hranice objektů a hledá na nich prohlubně. Výsledky jsou zapsány v Tabulce 1. Oba programy jsou spuštěny se stejnými parametry určující soliditu a velikost jednotlivých jader. Počet osamocených jader na snímku, které jsou vybrány před dělením bude stejný u obou programů. Rozdíly nalezených počtů jader se liší tedy v jádrech rozdělených ze shluků. Z výsledků zapsaných v tabulce můžeme usoudit, že počet chyb je u obou programů přibližně stejný. Program s metodou aktivních kontur dává více jednotlivých rozdělených jader, nalezl jich celkově na 22 snímcích 991. Program s prohledáváním hranic nalezl pouze 753 jader. Je zde zjevná nevýhoda programu s metodou aktivních kontur v časové náročnosti, pracuje 10 až 20 krát pomaleji. Výsledné časy u programu s metodou aktivních kontur jsou ale stále únosné. Navíc poskytuje ve výsledku mnohem více jader než druhý program s přibližně stejným počtem chyb, je tedy efektivnější.

	Program s metodou aktivních kontur			Program s prohledáváním hranic		
snímek	počet jader	počet chyb	čas [s]	počet jader	počet chyb	čas [s]
1	44	0	227	30	0	14
2	48	1 (2)	237	25	0	13
3	27	(1)	173	19	(1)	12
4	32	0	109	23	0	10
5	21	0	36	20	0	6
6	25	0	78	19	0	8
7	32	(1)	53	23	0	7
8	26	0	95	20	0	8
9	34	0	104	25	0	11
10	37	0	110	32	0	11
11	42	(1)	162	33	(1)	11
12	50	2	244	26	0	15
13	46	0	234	34	0	15
14	40	0	189	35	0	16
15	42	1	154	31	0	12
16	31	0	75	28	(1)	8
17	39	(1)	121	26	0	10
18	40	0	134	25	0	12
19	28	0	111	20	0	11
20	31	(1)	125	21	0	10
21	143	0	426	123	2 (3)	25
22	133	(1)	355	115	2 (4)	19
celkem	991	4(8)		753	4(10)	

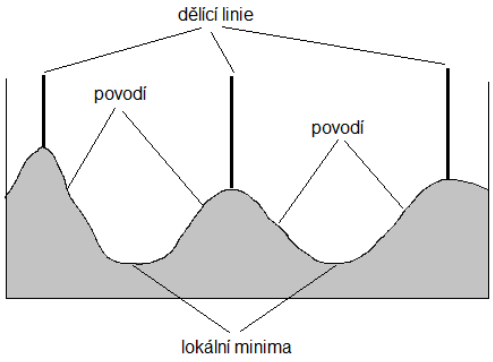
Tabulka 1: Výsledky programů při nastavení parametru solidity na 0.95. Počet jader udává počet nalezených a oddělených jednotlivých jader, které jsou uloženy ve výsledném snímku. Sloupec počet chyb ukazuje počet špatných řezů. Čísla bez závorek udávají počet špatných řezů způsobených programem, čísla v závorkách udávají chyby řezu, které jsou způsobeny tvarem jader. Sloupec čas udává trvání daných programů na jednotlivých snímcích při testování v sekundách.

6.4 Postup využívající metodu watershed

Další nápad pro tento problém využívá stejného principu jako postup s prohledáváním hranic, ale navíc se kombinuje s metodou watershed. Tento postup je ve fázi myšlenky a není ještě zcela realizován.

Watershed

Watershed je metoda pro automatickou separaci objektů. Nejdříve je nutné vypočítat vzdálenostní mapa binárního snímku. Pro každý pixel objektu se vypočte vzdálenost od nejbližšího bodu pozadí. Nejčastěji se používá klasická Euklidovská vzdálenost dvou bodů $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Tímto dostaneme snímek ve stupních šedé, který si můžeme představit jako reliéf. Když si představíme, že budeme reliéf zaplavovat padající vodou, můžeme rozlišovat tři typy bodů. Jsou to lokální minima. Dále jsou to body, na které pokud spadne voda steče do jednoho z minim, nazývané také povodí. Třetí typ bodů jsou body, ve kterých je stejně pravděpodobné, že voda steče do jednoho či druhého minima. Tyto body tvoří hřeben dělící různá povodí a znázorňuje dělící linii (viz [19]). Více o této metodě můžeme nalézt například v [20].

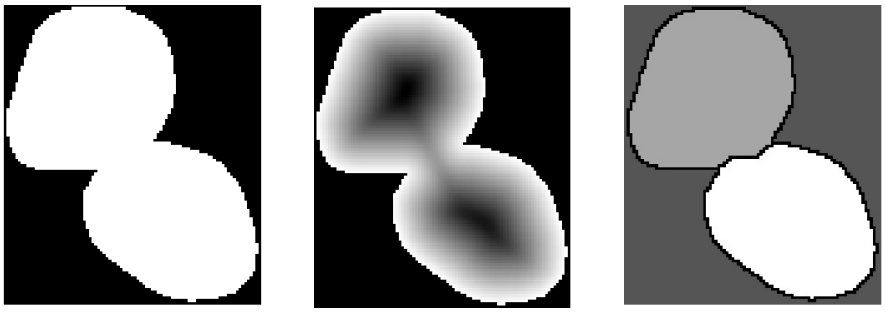


Obrázek 6.8: Watershred na jednorozměrném příkladě

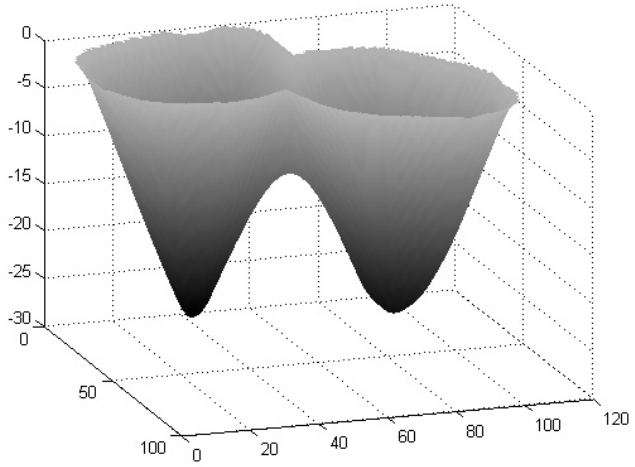
Program

Nejdříve je postup stejný jako v předchozím algoritmu. Rozdělí se hranice objektu a pro každý bod se vypočítá třetí složka vektorového součinu. Z těchto hodnot se nejdříve vyberou záporné hodnoty a mezi nimi se hledají lokální minima. To určuje body, ve kterých je úhel vektorů $\overrightarrow{u}$ a $\overrightarrow{v}$ správně orientovaný a nejvíce ostrý v porovnání s okolními body.

Jelikož je těchto bodů zpravidla více než míst, kde by se objekty měly správně dělit, tento postup je zkombinován s metodou watershed. Tato metoda je implementovaná v Matlabu, ale samotná nedává příliš dobré výsledky. Výsledkem watershedu je snímek rozdělen na oblasti, podle řezů určených touto metodou. Procházejí se postupně body z první části, ve kterých jsou prohlubně a zapíšou se oblasti z watershedu v jejich okolí. Pokud nějaké dva body mají stejné oblasti ve svém okolí a tyto oblasti jsou právě čtyři, tj. dvě jádra, pozadí a křivka (hranice), provede se mezi těmito dvěma body lineární řez.



(a) (b) (c)



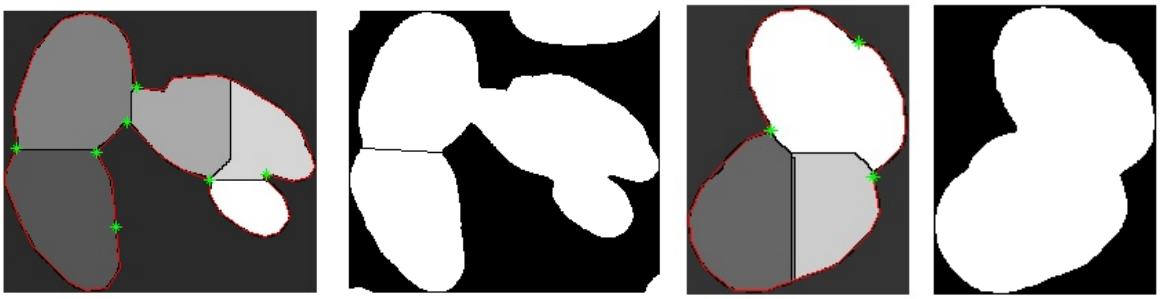
(d)

Obrázek 6.9: Postup watershredu. (a) binární snímek po prahování, (b) vzdálenostní mapa s přenastavenou hodnotou pozadí, (c) výsledek wateshredu, (d) vzdálenostní mapa vykreslená ve 3D

Nevýhody postupu s metodou watershed

U druhé metody se body párují k řezu, tak aby měly ve svém okolí stejné oblasti z metody watershed. Nastávají ale situace, jako na Obrázku 6.10. Body nemají stejné

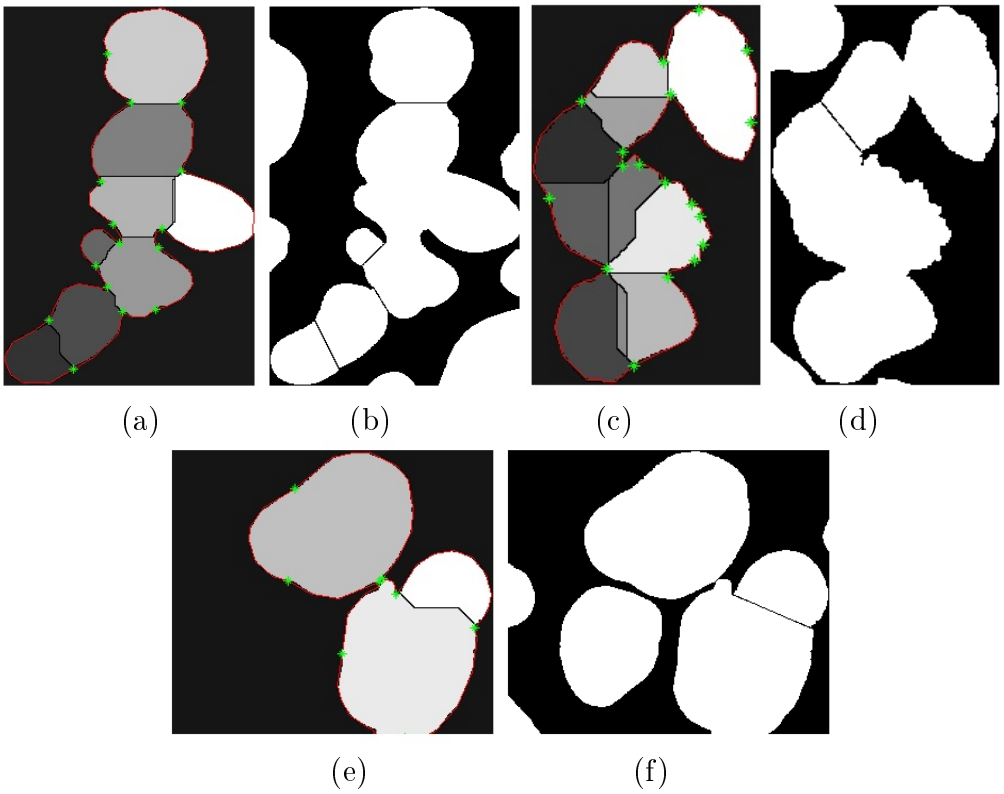
okolí, i když by očividně v těchto dvou bodech měl být veden řez. Je to způsobeno špatnými výsledky z watershedu nebo tím, že jeden z bodů leží v rohu další oblasti.



(a) (b) (c) (d)

Obrázek 6.10: Nevýhoda druhého programu, střet další oblasti v bodě, který je kandidát na řez. (a) odstíny šedé jsou rozděleny oblasti po watershredu, zeleně jsou označeny body, které jsou kandidáty na body řezu z hledání prohlubní na hranici. (b) objekt s výsledným řezem

Dále se mohou často objevovat „nesprávné“ řezy. Často se to může stát u podélných jader nebo u objektů které mají nějakou nerovnost. Příklady můžeme vidět na Obrázku 6.11. Některé případy by se daly vyřešit podmínkou na délku řezu. Další nevýhoda spočívá v tom, že kandidáti na body řezu nejsou dost blízko řezu z metody watershed. Tato nevýhoda programu by se z velké části dala vyřešit správným nastavením parametrů, jako je výběr bodů pro nalezení prohlubní a velikost okolí kandidátů na body řezu. Parametry je testováním nutno nastavit tak, aby se v takovýchto případech prováděly správně řezy, ale i tak, aby nepřibývaly nesprávné řezy.



Obrázek 6.11: Nesprávné řezy. (a) , (c), (e) výsledky z programu, plochy s různými odstíny šedé jsou oblasti po watershredu, zelené body jsou kandidáti na dělení z prohlubní, (b), (d), (f) výsledné řezy

Závěr

Práce je zaměřená na metody segmentace obrazu a na jejich praktické využití na konkrétním příkladu. V práci jsem nejprve uvedla jednu z nejjednodušších metod segmentace obrazu zvanou prahování. Zmínila jsem několik postupů pro nalezení prahu pro tuto metodu, včetně nejznámější Otsuovy metody. V další částí jsem se zabývala metodou aktivních kontur a to především metodou Chan-Vese. Ukázala jsem odvození metody a jeho modelu pomoci level set formulace a následný přechod k jeho numerické formulaci.

Hlavní cíl práce, vytvořit program, který by detekoval jednotlivá obarvená jádra pomoci DAPI a rozdělil shluky na jednotlivá jádra, se podařilo splnit. Napsala jsem program, který využívá metodu aktivních kontur pro nalezení bodů k řezu, který by oddělil jednotlivá jádra ze shluků. V práci jsem popsala podrobně jeho algoritmus a jeho parametry. Uvedla jsem také funkce Matlabu, které se využívají pro segmentaci obrazu a které využívám i ve svém programu. Dále jsem ukázala jaké jsou výsledky různých metod prahování a metody aktivních kontur při různých vstupních parametrech u daného typu snímků. Ve spolupráci s Tomášem Fürstem jsme vytvořily druhý program, který postupně prohledává hranici objektu a hledá zde prohlubně. V těchto prohlubních se poté vedou řezy. Následně jsem ukázala nevýhody obou metod a chyby, které se mohou objevit na konkrétních snímcích. Poté jsem aplikovala oba programy na sadě testovacích snímků a výsledky jsem porovnala. Oba programy vrací velice dobré výsledky s mnoha nalezenými a rozdělenými jednotlivými jádry a s málo chybnými řezy. Program s metodou aktivních kontur pracuje sice pomaleji, ale poskytuje o poznání více jednotlivých oddělených jader než program s prohledáváním hranice objektu.

Na závěr jsem uvedla myšlenku na další program na dělení shluků, který vychází z prohledávání hranice objektů jako předchozí program, ale navíc jej kombinuje s metodou watershed. Naznačila jsem možný postup a případné chyby, které by mohly nastat.

Při psaní práce a vývoji programu jsem se seznámila se základními postupy pro analýzu a segmentaci obrazu. Zdokonalila jsem se v této oblasti a to i v programovacím prostředí Matlab.

Literatura

[1] Canny, J., A Computational Approach To Edge Detection,IEEE Trans. Pattern Analysis and Machine Intelligence, 8(6), pp: 679–698, November 1986

[2] Wikipedia, Thresholding (image processing), http://en.wikipedia.org/wiki/Thresholding_(image_processing), [online], cit, 9.4.2015

[3] Kass, M., Witkin, A., Terzopoulos, D., Snakes: Active contour models, International Journal of Computer Vision, Volume 1, Issue 4, January 1988, pp 321-331

[4] Chan, Tony F.; Vese, Luminita A.; Active Contour Without Edges, Image Processing, IEEE Transactions on , vol.10, no.2, February 2001, pp.266,277

[5] Wikipedia, DNA repair, https://en.wikipedia.org/wiki/DNA_repair#DNA_damage, [online], 5.4.2015

[6] cancer.org, What Is Cancer?, http://www.cancer.org/cancer/cancerbasics/what-is-cancer, [online], cit. 5.4.2015

[7] Wikipedia, Confocal laser scanning microscopy, http://en.wikipedia.org/wiki/Confocal_laser_scanning_microscopy, [online], cit. 6.4.2015

[8] Ústav molekulární a translační medicíny Lékařské fakulty Univerzity Palackého, http://www.umtm.cz/ , [online] , cit. 6.4.2015

[9] Wikipedie, Distribuce (matematika), http://cs.wikipedia.org/wiki/Distribuce_(matematika), [online], cit. 3.4.2015

[10] Sezgin M., Sankur B., Survey over image thresholding techniques and quantitative performance evaluation, Journal of Electronic Imaging 13(1), 1. January 2004, pp 146-168

[11] Raju, P.Daniel Ratna, Neelima, G., Image Segmentation by Using Histogram Thresholding, IJCSET, January 2012, Vol 2, Issue 1, pp 776-779, ISSN:2231-0711

[12] Otsu Nobuyuki, A Threshold Selection Method from Gray-Level Histgrams, IEEE Transactions on Systems, Man and Cybernetics, Vol.9, No.1, pp 62-66, 1979

[13] Liao, Chen, Chung, A Fast Algorithm for Multilevel Thresholding, Journal of Information Science and Engineering 17, pp 713-727, 2001

[14] Mumford, David; Shah, Jayant; Optimal Approximations by Piecewise Smooth Functions and Associated Variational Problems, Communications on Pure and Applied Mathematics, Volume 42, Issue 5, July 1989, pages 577–685

[15] Blatter, Christian; Length of the zero level set of a function, Mathematics, [online], edited 11. April 2012, [cit. 4.3.2015], http://math.stackexchange.com/questions/129709/length-of-the-zero-level-set-of-a-function

[16] Li, C.; Xu, C.; Gui, C.; Fox, M.D.; Level Set Evolution Without Re-initialization: A New Variational Formulation, Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on , vol.1, no., vol. 1, 20-25 June 2005, pp. 430-436

[17] Rudin, L., Osher, S., Fatemi, E., Nonlinear total variation based noise removal algorithms, Physica D: Nonlinear Phenomena, Vol. 60, Issues 1–4, 1 November 1992, Pages 259-268, ISSN 0167-2789

[18] MathWorks, Matlab Documentation, [online], http://www.mathworks.com/help/matlab/

[19] Kaur, Amandeep; Aayushi; Image Segmentation using Watershed Transform, International Journal of Soft Computing and Engineering (IJSCE), Volume-4, Issue-1, March 2014, ISSN: 2231-2307

[20] Meyer, Fernand; Topographic distance and watershed lines; Signal Processing, Volume 38, Issue 1, , July 1994, Pages 113-125, ISSN 0165-1684

Přílohy

Součástí práce je přiložené CD s kódy obou programů, s testovacími snímky a výslednými snímky z obou metod.

Ve složce program s active contour je program s metodou aktivních kontur. Program se spouští skriptem spousteni.m, ve kterém se jako vstup zadává název snímku, který se má analyzovat. V tomto skriptu můžeme nastavit i parametry pro program. Výstupem je binární snímek jádro pouze s jednotlivými jádry

Ve složce program s prohledáváním hranic jsou uloženy skripty a funkce programu s prohledáváním hranic a složka data. Do složky data vložíme snímky, které chceme analyzovat. Program se spouští skriptem ZenNucleusFocus.m, kde se zadávají parametry. Skript si volá jednotlivé snímky ze složky data. Výstupem je binární snímek segment.mask s jednotlivými jádry.

Ve složce Testovací snímky jsou uloženy testovací snímky na kterých byly metody testovány.

Složky výsledky active contour a výsledky prohledávání hranic obsahují tři podsložky s výslednými snímky. Jedna složka obsahuje originální testovací snímky ve stupních šedé s barevně ohraničenými rozdělenými jednotlivými jádry. Druhá složka obsahuje snímky po prahování. Třetí složka obsahuje výsledné binární snímky, kde jsou pouze rozdělená jednotlivá jádra.