VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV BIOMEDICÍNSKÉHO INŽENÝRSTVÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF BIOMEDICAL ENGINEERING

VIZUALIZACE POVRCHU TKÁNÍ Z OBJEMOVÝCH OCT DAT

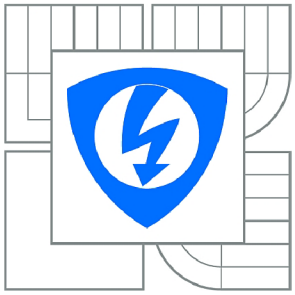
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MAREK KOSTIHA
AUTHOR

BRNO 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV BIOMEDICÍNSKÉHO INŽENÝRSTVÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF BIOMEDICAL ENGINEERING

VIZUALIZACE POVRCHU TKÁNÍ Z OBJEMOVÝCH OCT DAT

VISUALISATION OF TISSUE SURFACE FROM VOLUME OCT DATA

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

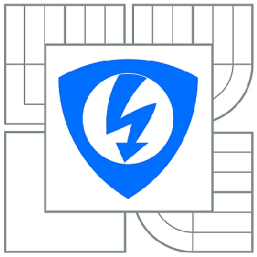
AUTOR PRÁCE — MAREK KOSTIHA

AUTHOR

VEDOUCÍ PRÁCE — Ing. VRATISLAV ČMIEL

SUPERVISOR

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav biomedicínského inženýrství

Bakalářská práce

bakalářský studijní obor
Biomedicínská technika a bioinformatika

Student:	Marek Kostiha	***ID:***	124994
Ročník:	3	***Akademický rok:***	2012/2013

NÁZEV TÉMATU:

Vizualizace povrchu tkání z objemových OCT dat

POKYNY PRO VYPRACOVÁNÍ:

1) Proveďte literární rešerši v oblasti akvizice obrazů OCT, jejich zpracování a analýzy se zaměřením na možnosti nalezení hranice povrchu tvrdé tkáně a jeho vizualizace. Seznamte se s Matlabem a jeho funkcemi práce s obrazem vhodnými pro tyto účely. 2) Zvolte metody pro nalezení a vizualizaci povrchu objektů z OCT obrazů. 3) Navrhněte systém pro akvizici a zpracování objemových OCT dat za účelem nalezní povrchu měřených objektů. 4) Realizujte návrh z bodu 3) v prostředí Matlab. Vytvořte grafické uživatelské prostředí pro načtení a zpracování dat. 5) Otestujte systém na OCT objemových datech pořízených na tvrdých tkáních s ostrým přechodem mezi vnitřní strukturou objektu a okolím objektu. 6) Proveďte diskusi nad získanými výsledky.

DOPORUČENÁ LITERATURA:

[1] ZAPLATÍLEK K., DOŇAR B. MATLAB tvorba uživatelských aplikací. Praha: BEN, 2004. ISBN: 80-7300-133-0
[2] GONZALES, R. C. Digital Image Processing. Prentice Hall, 2007. ISBN-13: 978-0131687288.

Termín zadání:	11.2.2013	***Termín odevzdání:***	31.5.2013

Vedoucí práce: Ing. Vratislav Čmiel
Konzultanti bakalářské práce:

prof. Ing. Ivo Provazník, Ph.D.
Předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Zobrazení povrchu je pro člověka téměř samozřejmostí. Využívá jasového rozdílu jednotlivých objektů a povrchu objektů. Použití v informatice je ovšem výrazně složitější. Moderní výpočetní technika se stále nedokáže svým výpočetním výkonem vyrovnat lidskému mozku. V dosažitelném maximálním rozlišení se ovšem lidské oko technice vyrovnat nedokáže. Přestože informatika a další technické obory jsou výrazně zatíženy šumem, který je zapotřebí potlačovat a některé technologie jsou velmi náročné, v mnoha směrech překonala možnosti člověka a stala se nepostradatelným pomocníkem.

Abstract

Display surface is almost commonplace for a man. It uses the difference in brightness of individual objects and surface objects. Use in science, however, is considerably more complicated. Modern computer technologies till not its computing power even the human brain. The maximum achievable resolution with the technology but the human eye cannot cope. Although science and other technical fields are significantly burdened by noise that is needed to suppress and some technologies are very challenging, in many ways surpassed human performance and has become indispensable.

Klíčová slova

Zpracování obrazu, šum, detekce hrany, vizualizace povrchu, Matlab, OCT.

Keywords

Image processing, noise, edge detection, surface visualization, Matlab, OCT.

KOSTIHA, M. *Vizualizace povrchu tkání pomocí objemových OCT dat*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií. Ústav Biomedicínské techniky a bioinformatiky, 2012. 21 s. Semestrální práce. Vedoucí práce: Ing. Vratislav Čmiel

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma Vizualizace povrchu tkání z objemových OCT dat jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářskou práci dále prohlašuji, že v souvislosti s vytvořením tohoto projektu jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 29. květen 2013 .....
podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Vratislavu Čmielovi. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne 29. květen 2013 .....
podpis autora

Obsah

1 Úvod

Vizualizace povrchu pokrývá široké spektrum využití od medicíny, přes mineralogii, archeologii. Lze ji použít všude tam, kde pracujeme s 3D objekty. Jediným omezením je použitá rozlišovací schopnost detektoru, které díky technologickému pokroku stále roste. Pro potřeby vizualizace povrchu lze použít elektromagnetické záření i mechanické vlnění (například ultrazvuk).

Optická koherentní tomografie (Optical Coherence Tomography, dále jen OCT), kterou se budu v této práci zabývat, je bezkontaktní neinvazivní diagnostická metoda, využívající elektromagnetické záření o vlnové délce ~ 1 mikron, uváděná rozlišovací schopnost je 4-20μm. Nejvíce je tato metoda používaná v oftalmologii, gastroenterologii a dermatologii.

Tomografický snímek je řez materiálem (například tkání). Nosič informace, kterým může být elektromagnetické záření nebo mechanické vlnění interferuje s materiálem, dochází k pohlcení části záření (případně vlnění) a tím změně intenzity, nebo odražení a tím změně fáze. Detektor, který může být součástí vysílače, detekuje změnu fáze nebo intenzity, počítač tyto změny zpracuje a vytvoří výsledný obraz. Detekce změny využívají například všechny typy rentgeny, detekce změny fáze využívá OCT, nebo ultrazvukové diagnostické přístroje.

Pro detekci hrany existuje již několik metod. Od nejjednodušších, založených na matematických operacích, až po složitější systémy neuronových sítí. Součástí detektoru hrany musí být i filtry, odstraňující nežádoucí šum, jelikož budeme zobrazovat neidealizované povrchy. Použití filtru je podmíněno citlivostí dané metody na šum a také typem šumu.

Na závěr úvodu bych shrnul jednotlivé kapitoly práce. V první kapitole se seznámíme s některými pojmy související s diskrétním obrazem, šumem a jeho odstraňováním. V další kapitole jsou uvedeny některé metody detekce hrany a jejich principy. Čtvrtá kapitola je věnovaná postprocessingu. V následující kapitole se již věnujeme samotné OCT technologii a jejímu principu. Předposlední kapitola se zabývá použitím detekce hran v prostředí Matlab. Poslední kapitola je již závěr této práce, ve kterém jsou shrnuty výsledky a plán dalšího průběhu bakalářské práce, která na tuto práci navazuje.

2 Počítačové zpracování obrazu

V této kapitole jsou uvedeny základní pojmy z oblasti zpracování obrazu a matematický operátor konvoluce. V další části kapitoly se budeme zabývat předzpracování obrazu.

2.1 Digitální obraz

Digitální obraz lze uložit dvěma způsoby. Prvním je bitmapová (rastrová) grafika, druhým je vektorová grafika.

Ve vektorové grafice je obraz složen z geometrických útvarů, jako jsou body, přímky, křivky atd. V této práci se nebudeme zabývat vektorovou grafikou, proto se zaměříme výhradně na rastrovou grafiku.

V bitmapové grafice jsou pixely (barevné body) uspořádány do mřížky. Každý pixel je určen svou polohou a barvu (v RGB obraze, černobílém obraze, obraze v odstínech šedi). Nevýhodou rastrové grafiky je omezení změny velikosti. Zvětšováním takového obrazu dochází ke ztrátě kvality. Při velkém rozlišení a barevné hloubce dosahuje obrázek velikosti až desítek MB. Mezi nejznámější formáty rastrového obrazu patří JPEG, PNG a BMP, liší se kompresí informací o jednotlivých pixelech.

Obraz lze definovat pomocí obrazové funkce:

$$z = f(x, y) \tag{2.1}$$

Pro práci s digitálním obrazem se často používá obraz v odstínech šedi, protože informace o barvě pixelu je pro většinu operací nepodstatná. Barevného obrazu se používá zvláště jako výsledku pro zvýšení informační hodnoty obrazu. Pro převod barevného obrazu se používá následujícího převodu:

$$I = r \cdot R + g \cdot G + b \cdot B \tag{2.2}$$

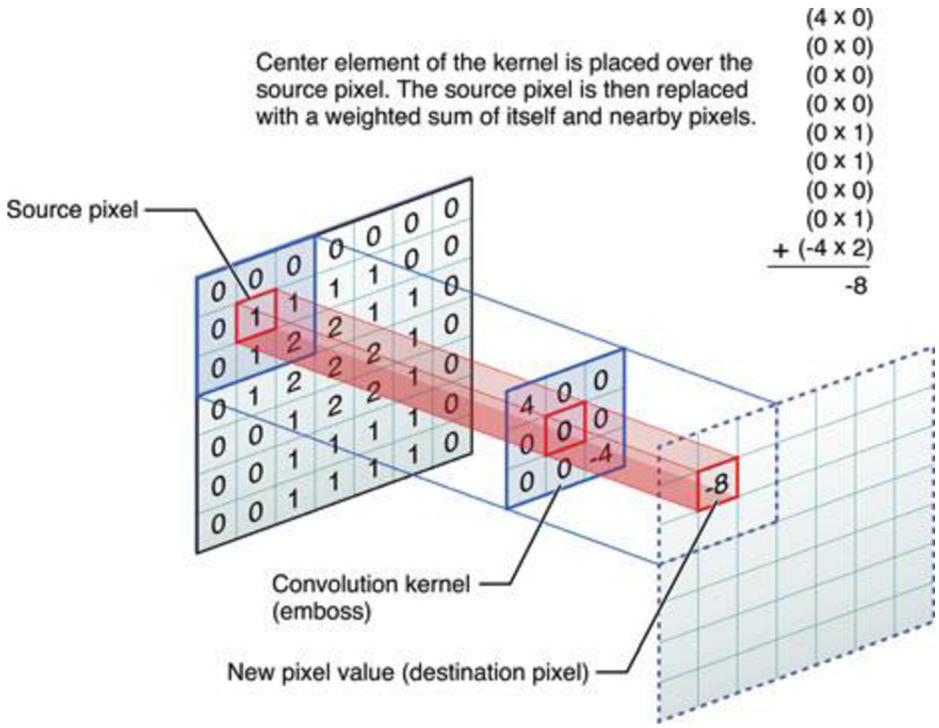
$$r + g + b = 1 \tag{2.3}$$

Indexy r, g a b představují citlivost oka na jednotlivé barevné složky. Obecně platí, že oko je citlivé nejvíce na zelenou barvu a nejméně na modrou barvu.

Základem filtrování obrazu a detekce hran je takzvaná konvoluce. Konvoluce je matematický operátor zpracovávající dvě funkce. Vzorec pro diskrétní konvoluci zpracovávající obraz má tvar:

$$(f * h)(x, y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f(x - i, y - j) \cdot h(i, j) \tag{2.4}$$

Konvolučním jádrem je konvoluční maska rozměrech $< -k, k > \times < -k, k >$. Výslednou hodnotu v každém bodě získáme součtem hodnot konvoluční masky vynásobených hodnotami vstupního obrazu. Princip konvoluce je uveden na Obrázku 2.1.



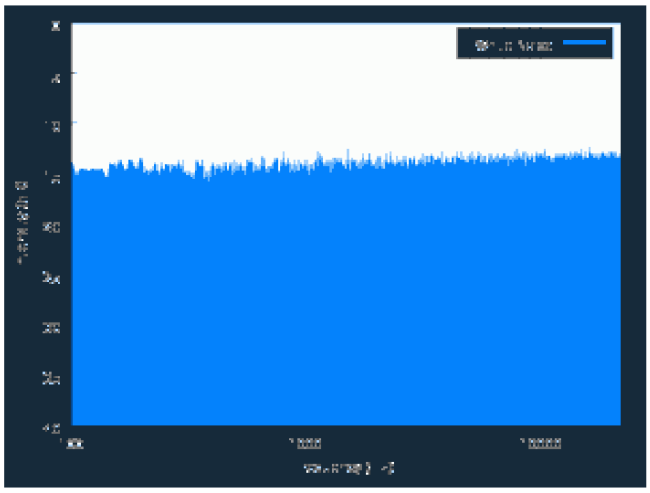
Obrázek 2.1: Princip diskrétní konvoluce (převzato z [1])

2.2 Předzpracování obrazu

Hlavním cílem předzpracování obrazu je odstranění šumu. Za šum můžeme obecně považovat jakoukoliv část signálu, která nějakým způsobem zakrývá důležitou informaci.

Rozlišujeme dva druhy šumu - aditivní a multiplikativní. Mezi základní typy šumu patří bílý šum, Gaussův šum a impulsní šum. Šum se v různé míře objevuje v každém reálném (neidealizovaném) měření, zpracování dat atd. Metody odstraňování šumu se liší jak podle typu šumu, tak i podle dostupných dat. V případě možnosti opakování měření a zajištění získání stejných výsledků můžeme například použít metody korelace. Této metodě se v této práci nebudeme věnovat, protože pravděpodobně nemůžeme zaručit shodnou akvizici dat. [3]

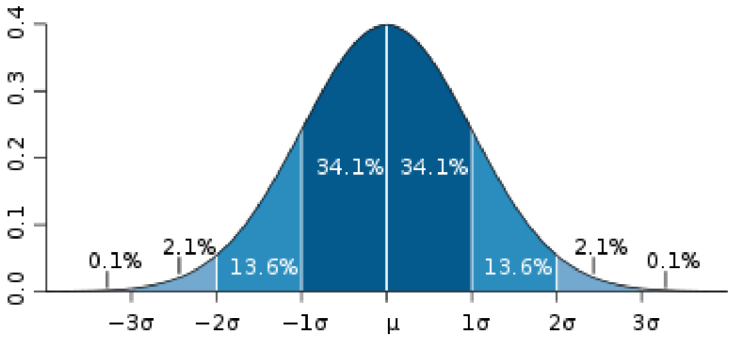
- Bílý šum - teoreticky má ve frekvenčním spektru rovnoměrně rozložen výkon mezi všechny frekvence (viz Obrázek 2.2). Ve skutečnosti je tento šum frekvenčně omezen, jinak by celkový výkon byl nekonečný. [3]



Obrázek 2.2: Spektrum bílého šumu (převzato z [4])

- Gaussův šum – pravděpodobnost výskytu tohoto šumu je dán Gaussovým rozdělením (viz Obrázek 2.3). Tato funkce má 2 parametry – střední hodnota **μ** standardní odchylka **σ**. Šum postihuje celý obraz.[3]

$$f(x) = \frac{1}{\sqrt{2\pi\sigma} \cdot e^{\frac{-(x-\mu)^2}{2\sigma^2}}} \tag{2.5}$$



Obrázek 2.3: Gaussovo rozdělení pravděpodobnosti ((Dostupné z WWW: [http://www.samouk.cz/psychologie/zaklady-statistiky/rozdeleni.png])

Impulsní šum – tento šum se projevuje v obraze impulsní (bodovou) změnou intenzity pixelů. Rozložení těchto pixelů je ve skutečnosti náhodné. Tento šum se často nazývá jako *Sůl a pepř* (Salt & Pepper). Příklad vlivu tohoto šumu je na obrázku 2.4. [3]

Obrázek 2.4: Příklad projevu impulsního šumu – 30% (převzato z [6])

Každý typ šumu vyžaduje poměrně specifický přístup. Nelze použít jednu metodu na jakýkoliv šum, protože by takový filtr nemusel šum odstranit a pravděpodobně by ještě více snížil kvalitu výstupního obrazu. Podrobněji jsou tu popsány tyto metody: Obyčejné průměrování, Gaussův filtr a medián.[3]

- Obyčejné průměrování – Nejjednodušší metoda odstranění šumu. Princip této metody spočívá na zprůměrování okolí bodu. Využívá konvoluční masky (viz Kapitola 2.1). Konvoluční maska musí mít lichý počet řádků a sloupců pro určení středového pixelu. Celková hodnota u všech konvolučních masek musí být rovna jedné, jinak by výsledný obraz byl světlejší nebo tmavší. Pro splnění této podmínky je před maskou konstanta. [3]

$$h_{prum} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{2.6}$$

- Gaussův filtr – tento filtr je také založen na použití vhodné konvoluční masky a slouží k odstranění Gaussova šumu. Konvoluční jádro není ovšem rovnoměrně rozloženo, ale objevuje se zde Gaussovo rozložení. V praxi se používá z technických důvodů již přepočítaných masek. Jako příklad je zde uvedena maska 5x5. [3]

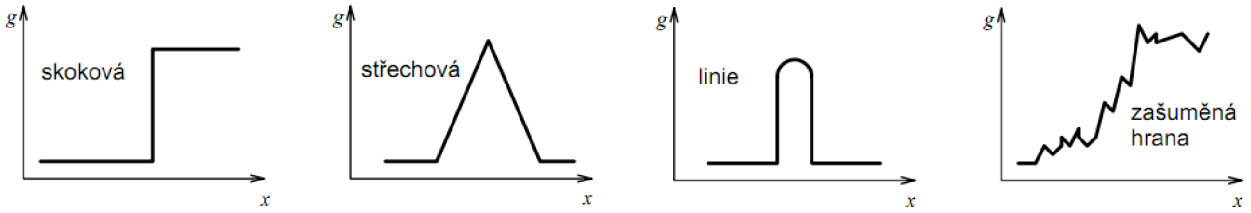
$$h_{gauss} = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} \tag{2.7}$$

- Medián – tento filtr nejúčinněji odstraňuje impulsní šum, navíc na rozdíl od konvolučních filtrů zachovává hrany v obrazu. Nevýhodou tohoto filtru při použití velkého okolí je potlačení detailů. Mediánový filtr vzestupně seřadí pixely a z této posloupnosti vybere prostřední (medián). V případě použití okna s lichým počtem prvků vypočítá průměr z prostředních dvou (nebo čtyř v případě dvourozměrného okna) pixelů. Tímto se naruší hrana a dochází k výrazné degradaci obrazu.

3 Detekce hrany

Tato kapitola je věnována samotné detekci hrany. Než však můžeme začít popisovat konkrétní metody, je nutné si definovat některé důležité pojmy, které se budou v této kapitole objevovat.

Nejprve si musíme ujasnit, co vlastně hrana v počítačovém obraze je. Hranou označujeme místa v obraze, kde prudce narůstá intenzita jasu sousedních pixelů. V případě dvoubitového (černobílého, ideálního) obrazu je tato změna skoková. Na velmi podobném principu pracuje i lidské oko, které takto rozeznává jednotlivé objekty (například jednotlivé písmena na papíru či monitoru). Míru, s jakou jas narůstá lze vyjádřit její derivací. Hrany lze dle průběhu rozdělit do několika tříd. Skoková odpovídá černobílému obrazu, střechová a impulsní jsou ideální hrany. Ve reálných obrazech nacházíme zašuměné hrany (obrázek 3.1). [3]



Obrázek 3.1: Typy hran (převzato z [2])

3.1 Metody založené na první derivaci

Jak bylo zmíněno v úvodu této kapitoly, hranu lze najít derivací funkce jasu v obraze. V diskrétním obraze získáme derivaci jako rozdíl pixelů v určitém okolí v obraze. Gradient neboli hodnota první derivace se provádí většinou ve dvou nebo osmi směrech. Existuje několik metod, které se liší použitým hranovým operátorem (konvoluční maskou). Mezi nejpoužívanější patří Robertsův operátor, Sobelův operátor a Prewittové operátor. Nejčastěji se používá maska 3x3, ale je nutné zvolit operátor a upravit velikost masky pro každý případ zvlášť. Nyní se seznámíme s konvolučními maskami jednotlivých operátorů (vzorce 3.1-3.3). [3]

Operátor	Řádkový gradient	Sloupcový gradient	
Robertsův operátor	$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$	(3.1)
Prewittové operátor	$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$	(3.2)
Sobelův operátor	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 1 & 0 \\ 1 & 2 & 1 \end{bmatrix}$	(3.3)

Výpočtem gradientu v osmi směrech můžeme zlepšit detekci hran. Masky pro výpočet gradientu ve dvou směrech jsou navzájem otočené o 90°, při výpočtu v osmi směrech jsou masky otočené o 45°. Výsledný gradient získáme zvolením gradientu s největší odezvou. [3]

$$G(x,y) = \max\{|G_1(x,y)|, \ldots, |G_m(x,y)|\} \tag{3.4}$$

Směr gradientu	Robinson	Kirsch	Prewitt	
H_1 0°	$\begin{bmatrix}1 & 1 & -1\\1 & -2 & -1\\1 & 1 & -1\end{bmatrix}$	$\begin{bmatrix}5 & -3 & -3\\5 & 0 & -3\\5 & -3 & -3\end{bmatrix}$	$\begin{bmatrix}-1 & 0 & 1\\-1 & 0 & 1\\-1 & 0 & 1\end{bmatrix}$	(3.5)
H_2 45°	$\begin{bmatrix}1 & -1 & -1\\1 & -2 & -1\\1 & 1 & 1\end{bmatrix}$	$\begin{bmatrix}-3 & -3 & -3\\5 & 0 & -3\\5 & 5 & -3\end{bmatrix}$	$\begin{bmatrix}0 & -1 & -1\\1 & 0 & -1\\1 & 1 & 0\end{bmatrix}$	(3.6)
H_3 90°	$\begin{bmatrix}-1 & -1 & -1\\1 & -2 & 1\\1 & 1 & 1\end{bmatrix}$	$\begin{bmatrix}-3 & -3 & -3\\-3 & 0 & -3\\5 & 5 & 5\end{bmatrix}$	$\begin{bmatrix}-1 & -1 & -1\\0 & 0 & 0\\1 & 1 & 1\end{bmatrix}$	(3.7)

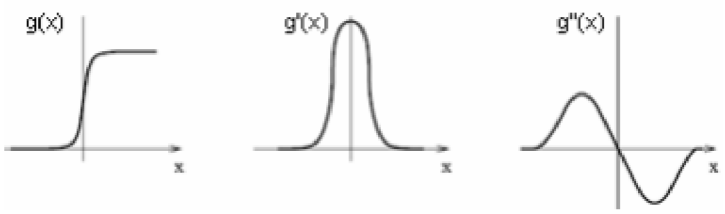
Dalším operátorem je první derivace Gaussovské funkce. Tento operátor kombinuje dva procesy, předzpracování pomocí Gaussova filtru a konvoluci s vhodným jádrem. Derivace i konvoluce jsou lineární operace, které lze libovolně zaměňovat. U tohoto operátoru se nejdříve vypočítá derivace vyhlazovací funkce a ta je poté konvolucí použita na obraz. Touto záměnou počítáme o jednu konvoluci méně. [3]

$$\frac{\partial}{\partial x}(h * f) = h * \frac{\partial f}{\partial x} = \frac{\partial h}{\partial x} * f \tag{3.8}$$

3.2 Metody založené na druhé derivaci

V případech, kdy potřebujeme znát pouze polohu hrany a nepotřebujeme znát velikost a směr hrany lze použít metody založené na druhé derivaci jasové funkce. Hrana v obrazu se nachází v místě, kde druhá derivace funkce prochází nulou. Pro detekování hran touto metodou existují dva způsoby. Můžeme použít dvojí výpočet první derivace pomocí rovnice 3.9, nebo použít některý hranový operátor počítající druhou derivaci. [3]

$$\Delta^2 f(x) = \frac{(f(x+1) - f(x-1))}{h^2} + O(h^2) \tag{3.9}$$



Obrázek 3.2: Průběh obrazové funkce: *a)původní funkce; b)první derivace; c)druhá derivace* (převzato ze [7])

Mezi nejpoužívanější operátory, založené na druhé derivaci, patří Laplaceovy operátory (viz vzorec 3.10). Součet všech prvků v operátorech se rovná nule a výrazné jsou středové body. [3]

Příklady Laplaceových operátorů:

$$\begin{bmatrix}0 & 1 & 0\\1 & -4 & 1\\0 & 1 & 0\end{bmatrix}\begin{bmatrix}1 & 1 & 1\\1 & -8 & 1\\1 & 1 & 1\end{bmatrix}\begin{bmatrix}-1 & 2 & -1\\2 & -4 & 2\\-1 & 2 & -1\end{bmatrix} \tag{3.10}$$

Laplaceovy operátory jsou podskupinou tzv. LoG (Laplacian of Gaussian). LoG používají jako konvoluční jádro přibližnou druhou derivaci Gaussova filtru. Výhodou LoG je možnost volby velikosti jádra, která má vliv na citlivost. Velikost tedy upravujeme podle velikosti detailů, které chceme detekovat. [3]

Příklad konvolučního jádra LoG:

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix} \tag{3.11}$$

Nevýhodou je náročnost výpočtu s rostoucí velikostí jádra. Tomuto se lze vyhnout použitím rekurzivních filtrů, nebo použít tzv. DoG filtru (Difference of Gaussian). DoG filtr použije na stejný obraz dvakrát Gaussův filtr pokaždé s jinou standardní odchylkou **σ**. Tyto dva obrazy od sebe následně odečteme. [3]

3.3 Cannyho hranový detektor

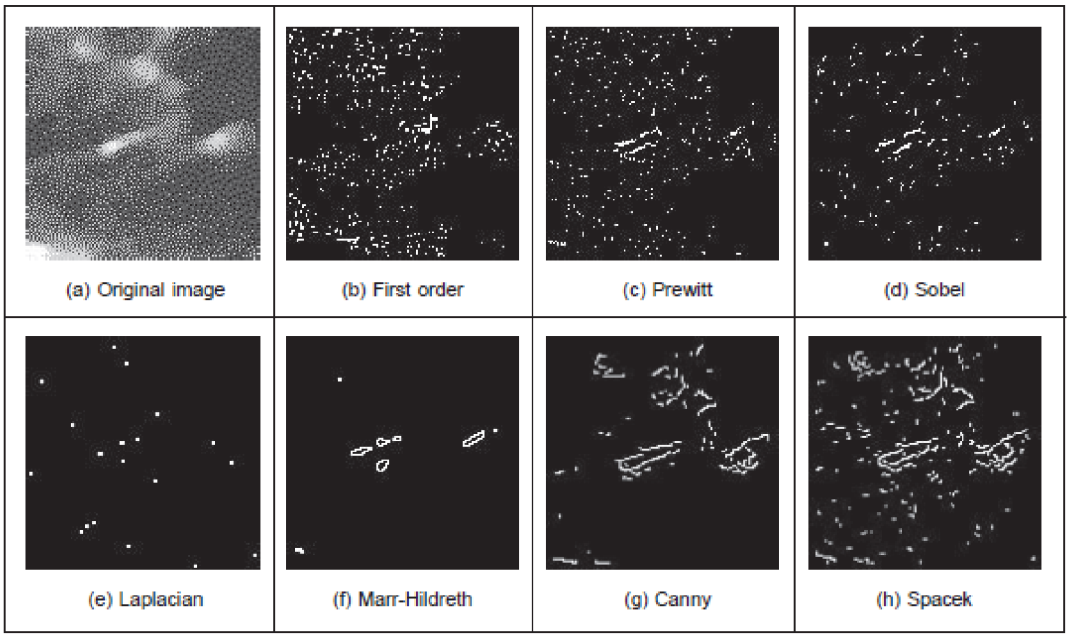
Téměř na pomyslném vrcholu mezi hranovými detektory stojí Cannyho detektor. Canny používá prahování s hysterezí pomocí dvou prahovacích hodnot(viz kapitola 4.2). Cannyho detektor je navržen tak, aby splňoval tři základní kritéria detekce hran:

- Minimální chybovost – musí být detekovány všechny hrany, které hranami skutečně jsou a nesmí být detekovány hrany, které hranami nejsou
- Lokalizace – detekovaná hrana musí být co nejblíže skutečné hraně
- Jednoznačná odezva – nesmí docházet k tzv. dvojité odezvě na hranu. Na jednu hranu smí připadat jen jedna odezva

Postup detekce hrany:

- Odstranění šumu – většinou se používá gaussův filtr
- Určení gradientu – vypočten pomocí konvoluce se Sobelovým operátorem nebo pomocí LoG
- Nalezení lokálních maxim – V gradientním obrazu jsou potlačeny hodnoty, které nejsou lokálními maximy
- Eliminace nevýznamných hran – gradientní obraz je převeden na binární obraz pomocí globálního prahování nebo prahování s hysterezí [3]

3.4 Srovnání jednotlivých operátorů



Obrázek 3.3: Srovnání různých metod detekce hrany (převzato z [5])

- Obrázek (a) – původní obraz, výrazně zašuměný
- Obrázek (b) – použití první derivace – odpovídá spíše na šum a je obtížné najít práh, který by odhalil skutečné hrany
- Obrázek (c) a (d) – objevuje se stále mnoho šumu, přesto lepší výsledek ukazuje Sobelův operátor
- Obrázek (e) – nedává téměř žádné informace o obrazu, ale pokročilejší operátory mohou přinést lepší výsledky
- Obrázek (f) – Marr-Hildreth (neboli DoG operátor): je obtížné zvolit vhodnou velikost operátoru tak, aby odstranil šum a zároveň detekoval hrany
- Obrázek (g) a (h) – Cannyho a Spackův hranový detektor – jednoznačně nejlepší výsledky, složitější aplikace v tomto případě má své opodstatnění [5]

3.5 Použití neuronové sítě

Pro detekci hran lze použít i neuronové sítě. Tyto neuronové sítě napodobují funkci biologického nervového systému, jsou tedy založeny na propojené síti neuronů. Umělá inteligence (dále jen UI, patří sem kromě neuronových sítí i fuzzy logika a další) se používá pro řešení velmi komplikovaných a obtížných případů, které nelze řešit běžnými metodami. Dovoluje totiž toleranci nepřesných dat nebo jejich aproximaci. Neuronové sítě jako detektoru hrany jakou součást složitějších problémů jako je rozpoznávání objektů v obraze, segmentace obrazu a podobně. Nepředpokládám, že bude tuto metodu nutné použít pro vizualizaci povrchu, proto zde nebude podrobněji rozpracována. [3]

4 Post-processing obrazu

Mezi procesy post-processingu patří algoritmus ztenčení hran a prahování. Tyto procesy se provádí po filtraci a detekování hran. [3]

4.1 Ztenčení hran

Algoritmus ztenčení hran (ang. Nonmaxima Suppresion) vybírá pouze lokální maxima z hodnot gradientu vypočítaného pomocí konvoluce s hranovým operátorem. Ostatní hrany jsou potlačeny. Algoritmus prochází bod po bodu a vybírá nejvyšší gradient v okolí, který je proti směru gradientu. [3]

Směr gradientu vypočítáme podle vzorce 4.1.

$$\theta(x,y) = tarcan\frac{G_y(x,y)}{G_x(x,y)} \tag{4.1}$$

4.2 Prahování

Nejjednodušším typem prahování je globální prahování. Používá jediný parametr **T** (treshold). Ke správnému zvolení prahu je vhodné využít histogramu obrazu. Prahováním obrazu potlačíme všechny hrany, které mají menší sílu (odezvu) než práh T. [3]

Pokročilejším typem prahování je takzvané prahování se dvěma prahy (neboli hysterezní prahování). Využívá dvou parametrů T_1 a T_2, kde $T_1 > T_2$ (v praxi se volí T_2 jako 1/3 – 1/2 T_1. Hrany, které jsou silnější než T_1 označujeme za hranu vždy, hrany silnější než T_2 jen za předpokladu, je-li nejméně jeden sousední pixel silná hrana. [3]

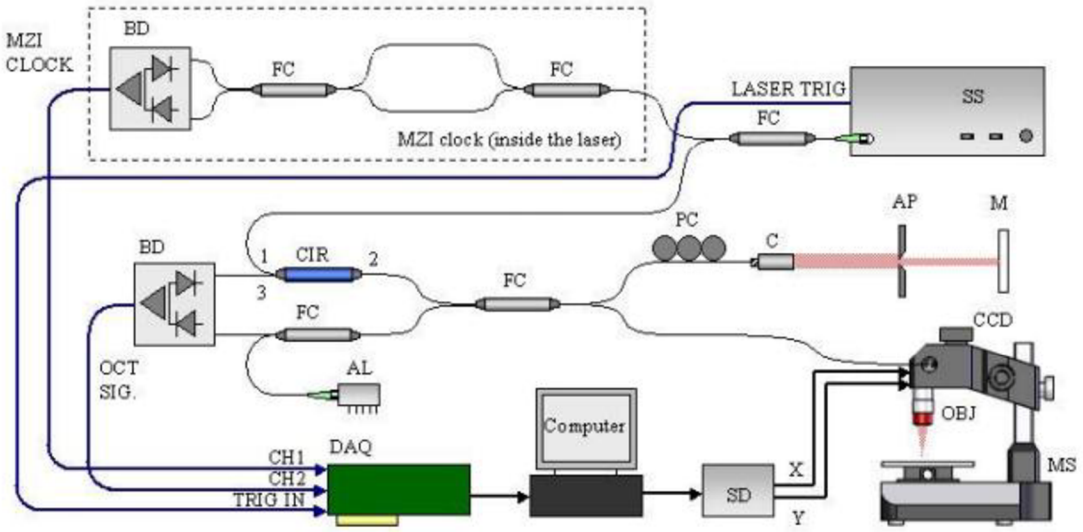
5 Optická koherentní tomografie

Optická koherentní tomografie (Optical Coherence Tomography, dále jen OCT), je bezkontaktní neinvazivní diagnostická metoda, využívající elektromagnetické záření o vlnové délce ~ 1 mikron, uváděná rozlišovací schopnost je 4-20µm. Nejvíce je tato metoda používaná v oftalmologii, gastroenterologii a dermatologii. OCT v klinické praxi používá dva identické paprsky. Jeden je odkloněn a slouží jako referenční paprsek, druhý je vysílán do vyšetřované tkáně. Ve vyšetřované tkáni dochází k lomu a rozptylu světla. Dále dochází k interferenci mezi původně téměř identickými paprsky, které jsou měřeny interferometrem. Z výsledných hodnot vytváří optický koherentní obraz. [8]

5.1 Teorie OCT

Swept source optical coherence tomography (SSOCT) technologie využívá rychle laděný úzkopásmový zdroj k osvětlení interferometru a zaznamenává informace jediným fotodetektorem. SSOCT technologie, jako je optická reflektometrie frekvenční oblasti, měří rozsah a časové zpoždění odraženého světla za účelem vytvoření hloubkových profilů (A-skenů je) vzorku, který je zobrazován. Jednotlivé A-scany (vytvořeno ve 2D módu, viz dále) jsou následně sloučeny k vytvoření celkového obrazu. Pokročilé metody sběru dat a digitálního zpracování signálu je součástí SSOCT systému umožňující v reálném čase zobrazovat obraz. Tento systém umožňuje vytváření OCT snímků podobných těm získaných například konfokálním mikroskopem sečtením signálů v axiálním směru. Vysokorychlostní 3D zobrazování umožňuje komplexní data, která v sobě spojují výhody povrchové mikroskopie a strukturální zobrazování struktury v jednom systému. [10]

Na Obrázku 5.1 je zapojení schéma zapojení Thorlabs OC S1300 SS



Obrázek 5.1: Schéma zapojení SSOCT systému (převzato z [10])

Tento systém obsahuje vysokorychlostní měnič frekvence laseru, který má spektrální šířku pásma 3 dB (větší než 100 nm) a průměrný výstupní výkon 10 MW. Zdroj má vestavěný Mach-Zehnder interferometr (MZI, Thorlabs INT-MZI-1300), který umožňuje frekvenční spínání laseru. Hlavním výstup laseru je připojen optickými vlákny k Michelsonovu interferometru. Na referenční a vzorkovací vlákno se paprsek rozdělí pomocí širokopásmového 50/50 zařízení (Thorlabs INT-MSI-1300). V referenčním vláknu interferometru se paprsek odráží zpět do vlákna pomocí pevného

zrcadla. Ve vzorkovacím vláknu vede světlo sondy, dále je paprsek usměrněn pomocí kolimátoru a přes skenovací zrcadla směřuje paprsek přímo na vzorek. Axiální skenování (A-scanů) jsou prováděny na 16 kHz. Příčné skenování (B-scan) je ovládáno pomocí skenovacích zrcadel a určuje počet snímků za sekundu. Pracovní vzdálenost mezi optikou a vzorkem je ≥25 mm. Vzorek je umístěn v držáku, který umožňuje posun v osách X a Y a rotaci vzorku. Integrovaná CCD kamera umístěná vedle objektivu umožňuje konvenční mikroskopický pohled na vzorek a tím pomáhá srovnat vzorek. [10]

5.2 Akvizice OCT dat

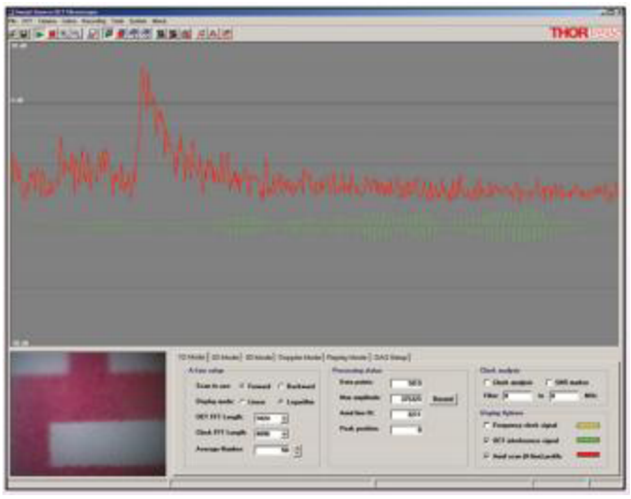
Interferující signál je detekován pomocí fotodetektoru, který potlačuje stejnosměrnou složku a autokorelační šum. Vysokorychlostní 14 bitový snímač sbírá vzorky OCT signálů, které jsou nejdříve pomocí rychlé Fourierovy transformace (FFT) převedeny z časové na frekvenční závislost. A poté kalibrovány. FFT interferujícího signálu obsahuje profil odrazivosti závislého na hloubce. Všechny požadované procesy sběru a zpracování dat probíhá prostřednictvím integrovaného softwarového balíku. [10]

5.3 Thorlabs OC S1300 SS

Tento model OCT přístroje bude použit pro akvizici objemových dat, pomocí kterých bude vizualizován povrch tvrdé tkáně. Přístroj má vlastnosti vhodné k zobrazování struktur i pod povrchem.

Přístroj dokáže pracovat v několika módech:

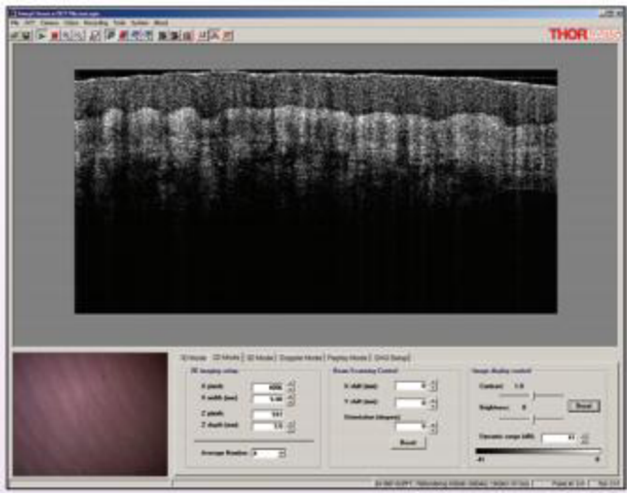
- 1D mód (A-scan zobrazení) – rameno s laserem se nepohybuje, Zobrazování v reálném čase kalibrovaných interferenčních proužků a Fourierova transformace rozptylové funkce (PSF, point spread function) pomáhá k optimalizaci signálu a nastavení parametrů. V tomto módu může uživatel ovládat dynamický rozsah a nastavení zesílení. Uživatel také může ukládat obrázky a naměřené hodnoty. [9]



Obrázek 5.1: Výstup při použití 1D módu (převzato z [9])

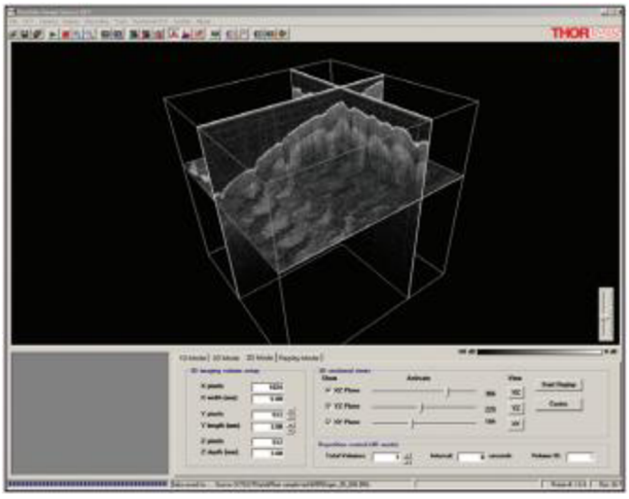
- 2D mód (B-scan) – v tomto zobrazovacím módu se rameno pohybuje v jednom směru a řezy se zobrazují na obrazovce v reálném čase. Software umožňuje úpravu

parametrů jako je vzdálenost, úhel snímání a počet řádků ve snímku. Uživatel také může upravovat jas a kontrast. [9]



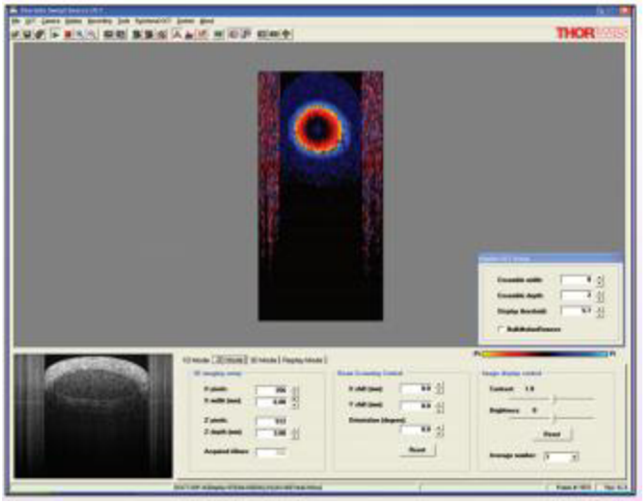
Obrázek 5.2: Výstup při použití 2D módu (převzato z [9])

- 3D mód – přístroj nasnímá celým vzorkem sérii snímků ve 2D módu, ze kterých následně sestaví 3D obraz vzorku. Uživatel může ovládat parametry X a Y pro objemovou akvizici. Celý model můžeme zobrazit v rovinách XY, XZ a YZ nebo v kombinaci těchto rovin. 3D model je uložen v naměřeném setu 2D rovin. [9]



Obrázek 5.3: Výstup při použití 3D módu (převzato z [9])

- Doppler Mód – V tomto módu jsou fázové posuny sousedních pixelů průměrovány pro výpočet změny frekvence vyvolané pohybem částic (obrázek 5.4). [9]



Obrázek 5.4: Výstup při použití módu Doppler (převzato z [9])

V následující tabulce (Tabulka 5.1) je uvedeno několik základních parametrů.

Střední vlnová délka	1325 ± 15 nm
Spektrální šířka pásma	120 ± 10 nm
Osové rozlišení (vzduch/voda)	12µm / 9 µm
Postranní rozlišní	15 µm
Maximální zobrazitelná hloubka	6,0 mm
Maximální počet pixelů (A-scan)	1024
Optický výkon na vzorku	5,0 mW

Tabulka 5.1: Základní parametry Thorlabs OC S1300SS (převzato z [9])

6 Detekce hran a prostředí Matlab

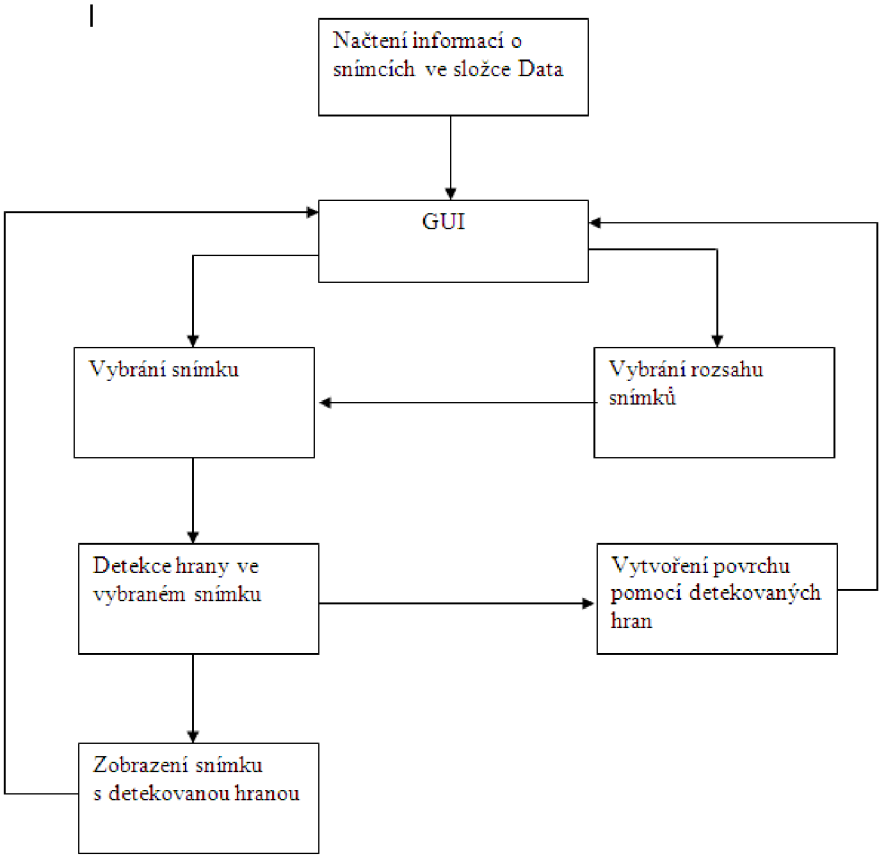
V této předposlední kapitole se seznámíme s prostředím Matlab a zpracováním obrazu v prostředí Matlab. V další části kapitoly je naznačen postup zpracování získaných snímků a v závěru této kapitoly jsou zveřejněny první výsledky detekování hran.

6.1 Zpracování obrazu v prostředí Matlab

Na úvod je třeba zmínit, že prostředí Matlab pracuje se všemi proměnnými jako s maticemi. Zároveň nezáleží na tom, zda současná proměnná byla původně obrázkem v jakémkoliv formátu, nebo pouze číslem či několika body křivky. Obraz lze snadno konvertovat na různé typy obrazu a lze provádět mezi proměnnými (maticemi) téměř libovolné matematické operace. Jádrem prostředí Matlab je knihovna funkcí, která velmi usnadňuje a zjednodušuje práci. Není tedy nutné si každou operaci programovat. Navíc ať už samotné naprogramování a ladění programu, tak i samotné výpočty jsou zpravidla výrazně pomalejší než při použití funkce z knihovny.

6.2 Postup zpracování snímků

Jak bylo zmíněno výše (viz kapitola 5.3), při snímání ve 3D módu se model uloží v jednotlivých řezech jako rastrové obrázky. Dalším krokem je načtení snímků, které tvoří nasnímaný povrch. Dále následuje odstranění šumu a nedůležitých částí obrazu (horní a dolní okraj). Důležitost odstranění šumu závisí na použité metodě detekce hran. Vybrání a použití metody v prostředí Matlab bude součástí bakalářské práce, která na toto téma navazuje praktickou částí. Následuje post-processing obrazu a nakonec se jednotlivé detekované hrany složí do jednoho 3D objektu, třetí ose bude vzdálenost mezi jednotlivými řezy tak, aby byly zachovány původní poměry stran.



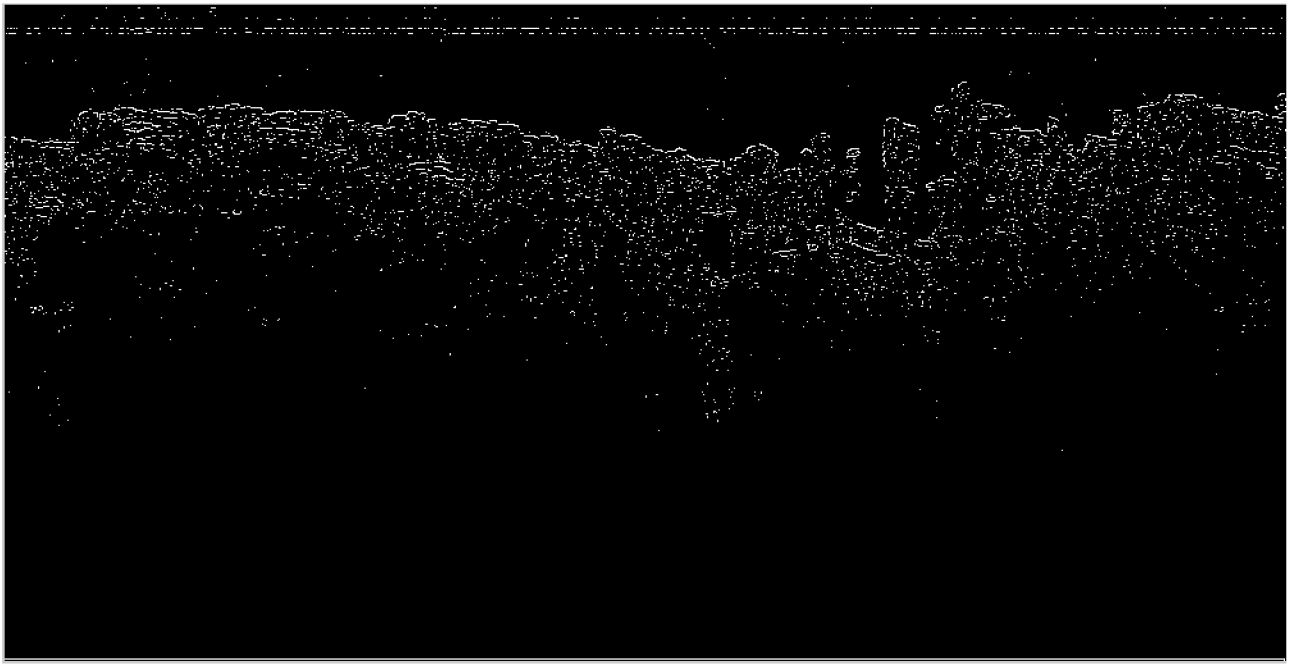
Obrázek 6.1: Blokové schéma procesu vizualizace povrchu v grafickém prostředí GUI Matlab

6.3 Metody detekce hrany

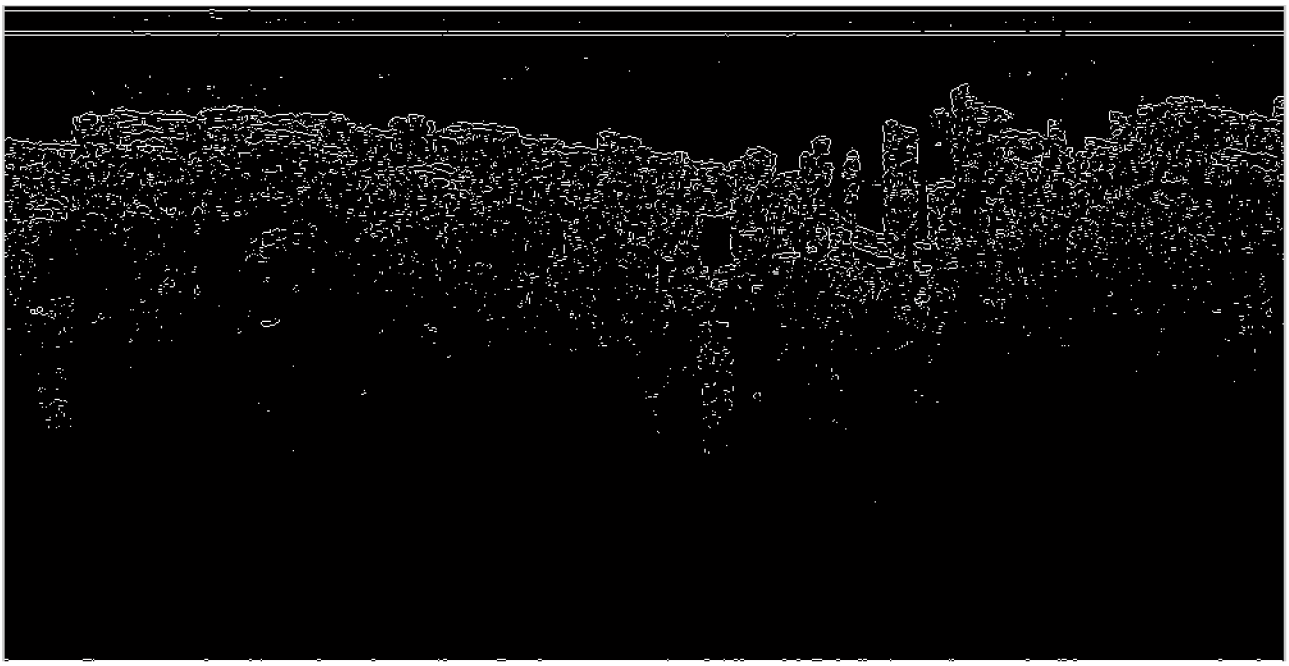
Během vypracovávání této práce jsem narazil a pokusil se využít několik způsobů detekce hrany. Nejjednodušší je metoda „Padající kapka“. Název této metody není oficiální. Jedná se o metodu, kdy je snímek procházen pixel po pixelu a je detekován nejvyšší pixel v daném sloupci, který překročí prahovou hodnotu. Tento pixel je považován za hranu obrazu. Tato metoda je velmi náchylná na šum a artefakty, je proto nutné použít filtrů a zvolit vhodnou prahovou hodnotu. Pokud v daném sloupci žádný z pixelů nepřekročí prahovou hodnotu, jako hrana je detekován nejvrchnější či nejspodnější pixel (v závislosti na konkrétním přístupu).

Jako další metodu detekce hrany jsem použil hranové operátory, které jsem již zmiňoval v kapitole 3. Knihovna funkcí Matlab-u nabízí několik metod. V rámci testování byly testovány tyto operátory: Sobel, Prewitt, Roberts, Laplacian a Canny. Nejlépe se osvědčil Cannyho operátor, protože krom spolehlivé detekce hrany také velmi spolehlivě odstraňuje šum v obraze. Na následujících snímcích budou představeny výsledky detekce hrany pro různé operátory pro srovnání. Výstupy těchto metod jsou na obrázcích 6.2-6.

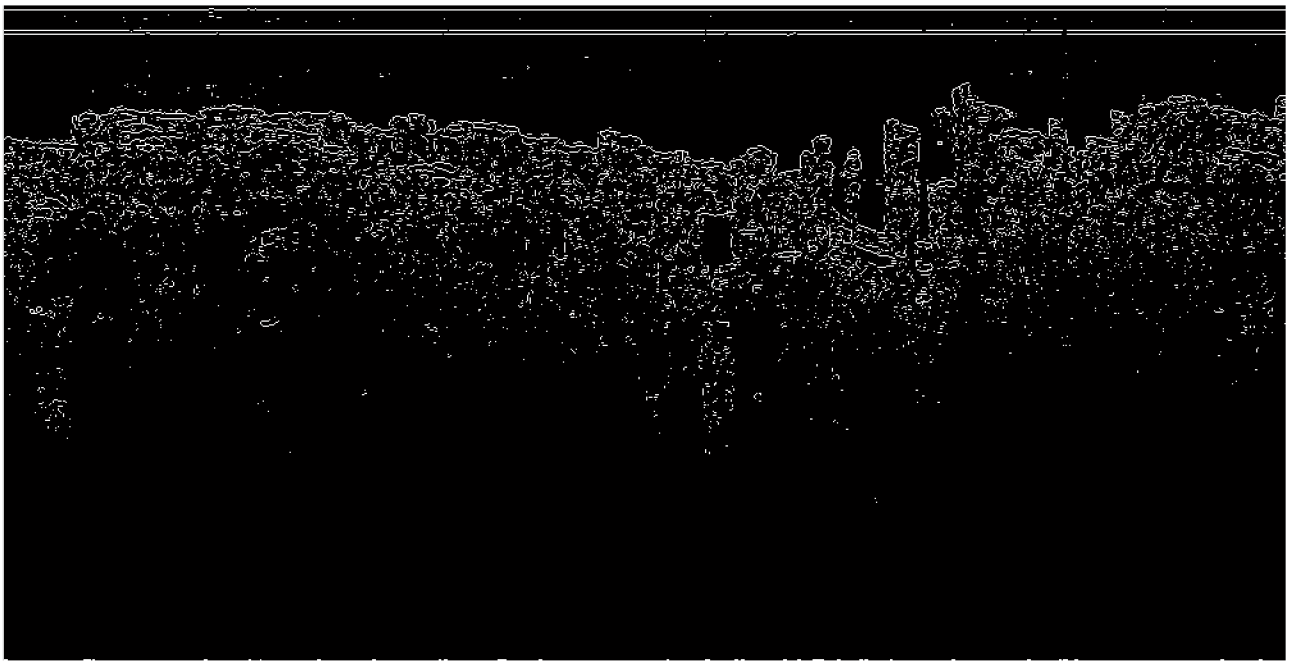
Jako poslední metodu jsem použil funkci *bwboundaries*. Tato metoda detekuje hranu v černobílém obrazu. Dokáže tak jednoduše určit souřadnice hrany a dokonce i dutin v obraze. Tato metoda nebyla kompatibilní s dalšími funkcemi. Výstupem této funkce jsou souřadnice detekovaných hran v obraze.



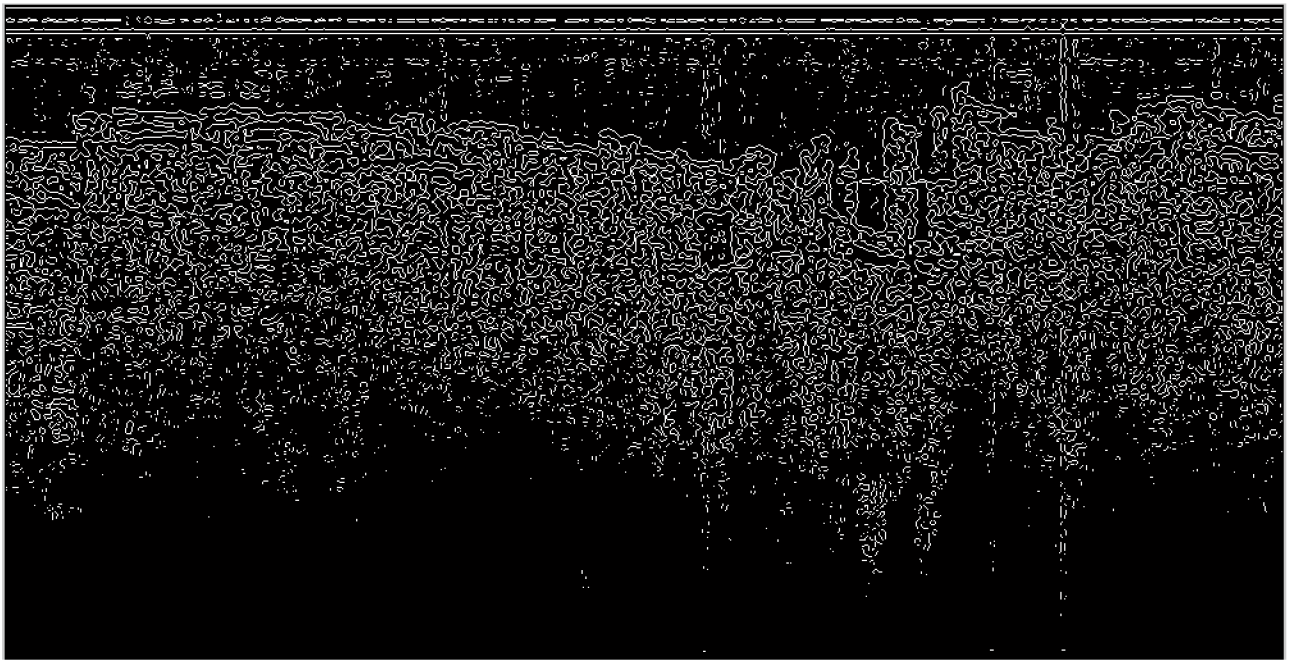
Obrázek 6.2: Hrany detekované pomocí Robertsova hranového detektoru



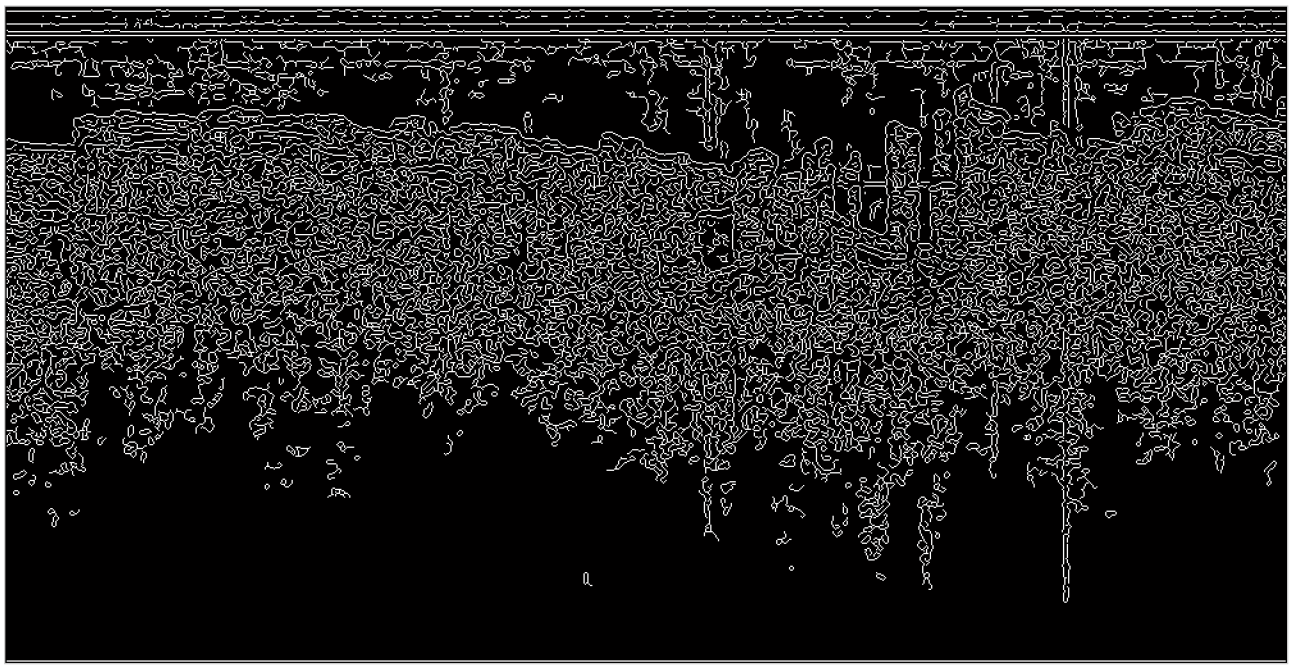
Obrázek 6.3: Hrany detekované pomocí Prewittova hranového detektoru



Obrázek 6.4: Hrany detekované pomocí Sobelova hranového detektoru



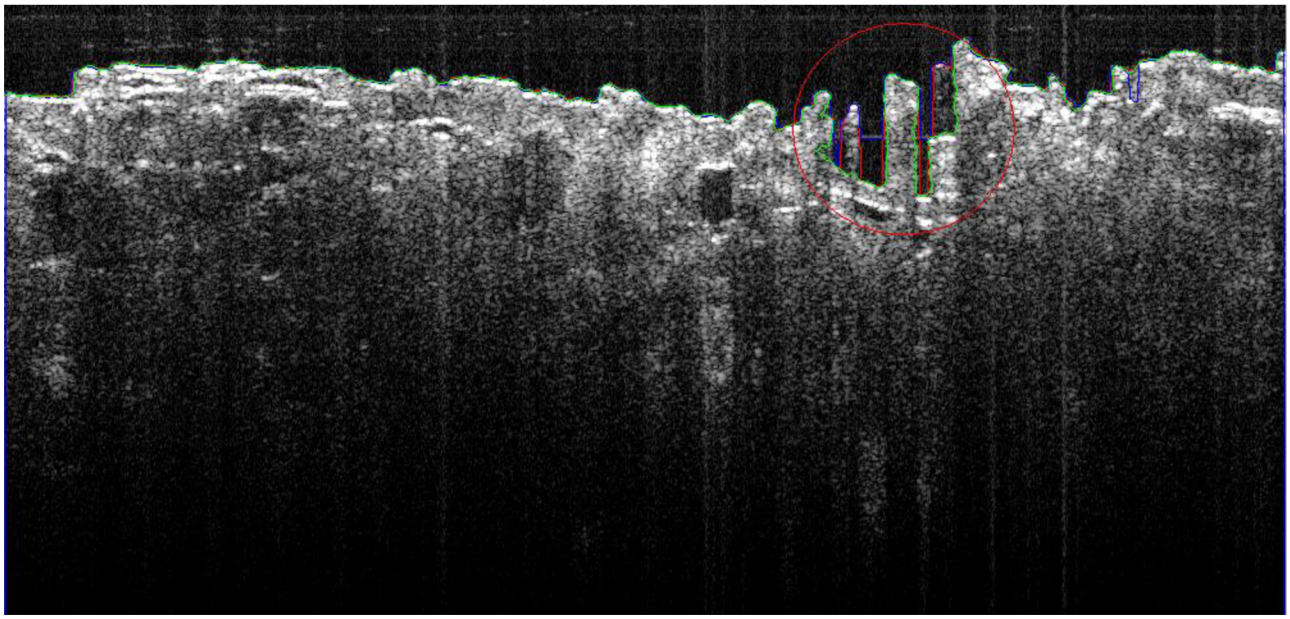
Obrázek 6.5: Hrany detekované pomocí Laplaciánového hranového detektoru



Obrázek 6.6: Hrany detekované pomocí Cannyho hranového detektoru

6.4 Srovnání výsledků detekce hrany různými metodami

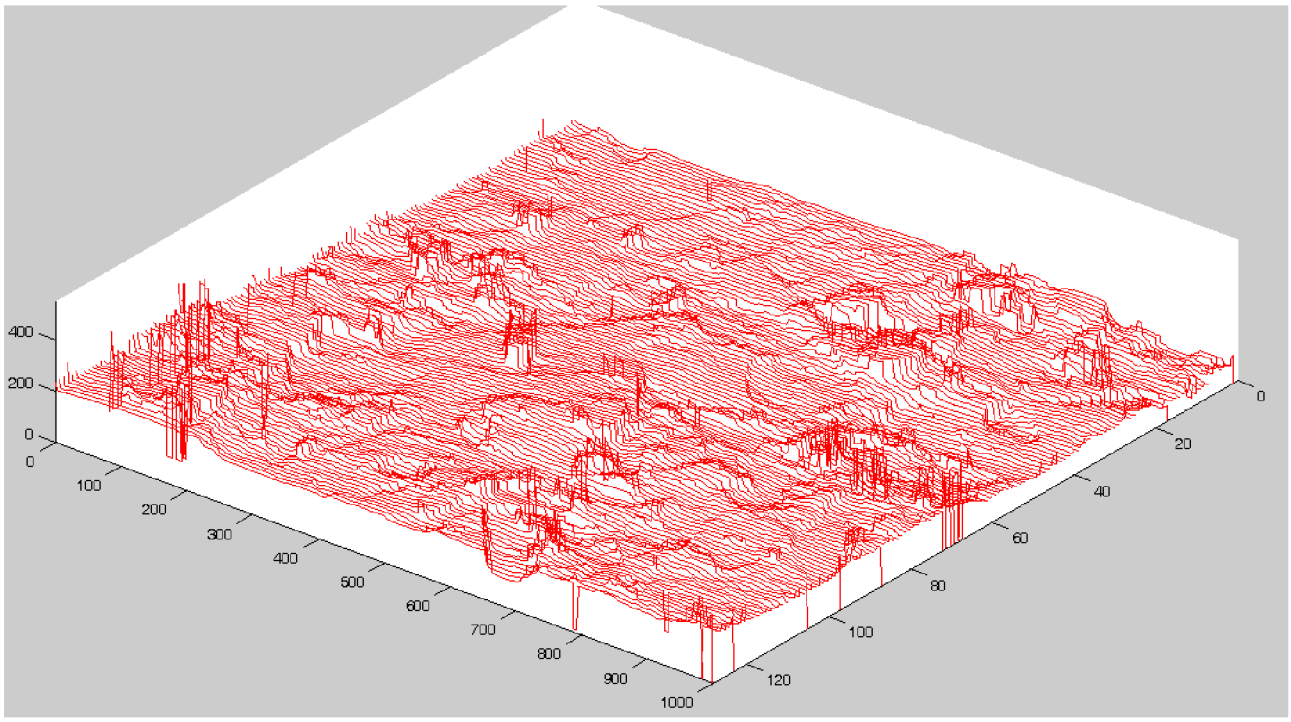
Na následujícím snímku je použito všech tří metod detekce, které jsem používal. Červená linie znázorňuje hranu detekovanou metodou „padající kapka“, zelená představuje výsledek použití funkce *bwboundaries* a modrá linie zvýrazňuje detekovanou hranu Cannyho operátorem. Ani jedna z metod není 100% úspěšná. Padající kapka nedetekuje nejsilnější hranu, funkce *bwboundaries* detekuje jen souvislou hranu a Cannyho operátor některé skutečné hrany nedetekuje nebo detekuje jako hranu artefakt. Celkově ovšem je účinnost Cannyho operátoru vyšší než u ostatních dvou metod.



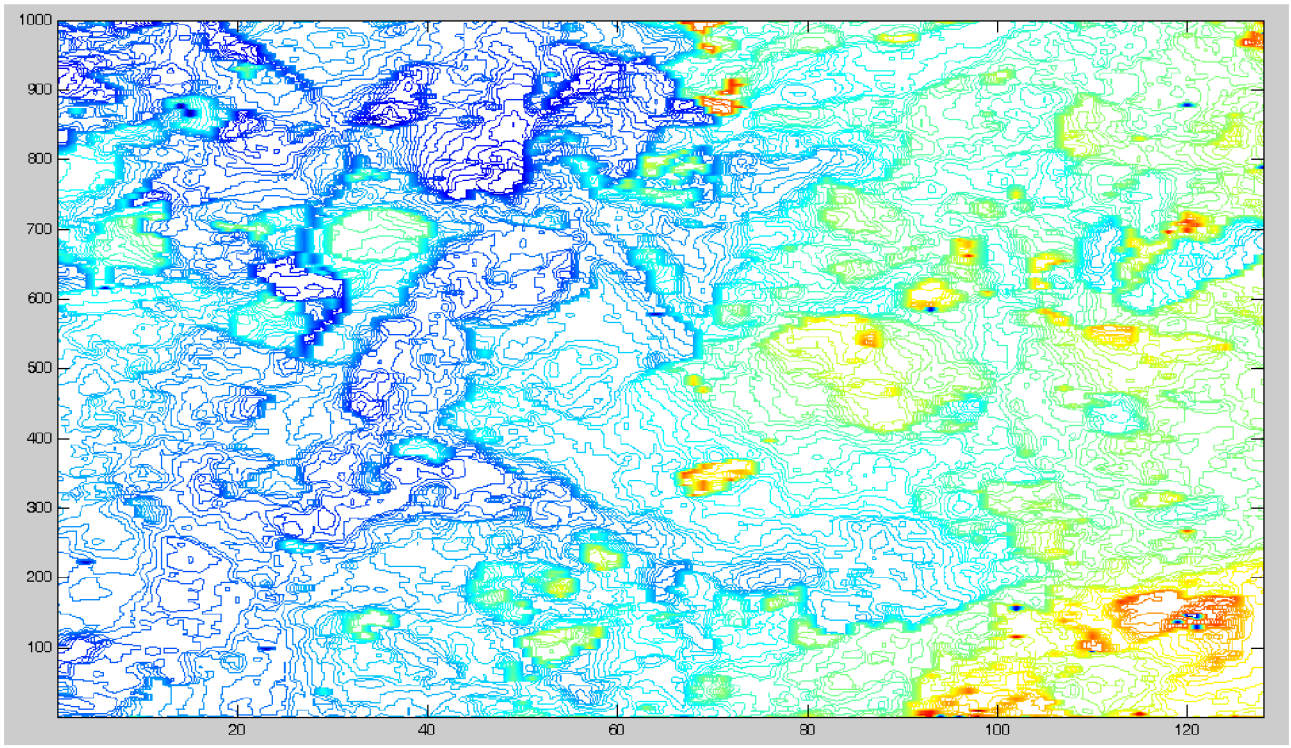
Obrázek 6.7: Detekované hrany („Padající kapka“ (červená), funkce bwboundaries (zelená) a Cannyho operátor (modrá))

6.5 Použité metody vizualizace povrchu

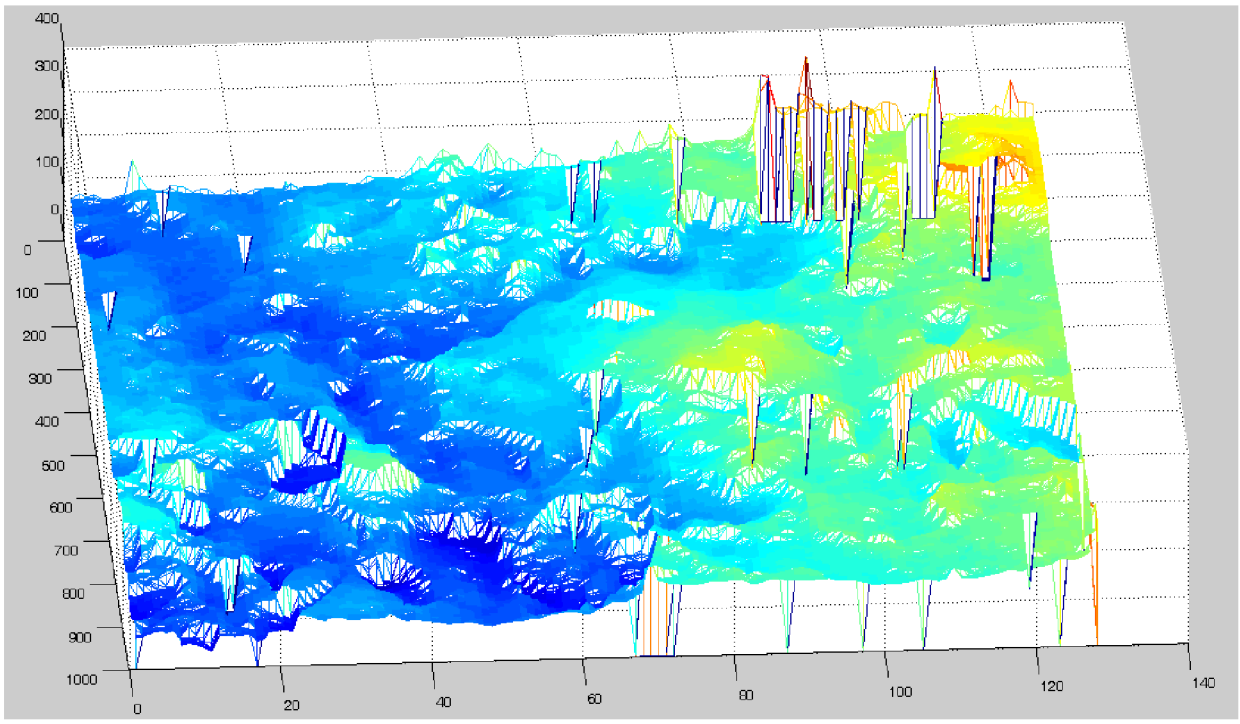
Matlab nabízí několik metod, jak zobrazit objemová data. Z jednotlivých řezů lze pomocí funkcí vytvořit matici, obsahující všechny souřadnice nalezených bodů hran každého snímku, které jsou nutné k vizualizaci nasnímaného povrchu. Nejjednodušším způsobem je zobrazení prostorové matice, ve které jsou vedle sebe s určitými mezerami naskládány jednotlivé nalezené hrany snímku. Tato metoda má tu výhodu, že dovoluje poměrně rychle odhalit množství chybně detekovaných hran (zvláště u vertikálních píků, způsobený artefakty). Další metodou je použití funkce *contour*, která vytvoří obraz podobný vrstevnicím na turistických mapách. Funkce spojí body, které odpovídají několika určitým rozsahům úrovní. Výška těchto bodů je znázorněna odstínem některé barevné mapy, a čím blíže jsou jednotlivé vrstevnice, tím je hrana strmější. Výsledkem této funkce je tedy 2D obraz s barevnými křivkami. Metodou s hodnotnějším výsledkem je funkce *mesh*. Tato funkce spojí detekované souřadnice hran všech snímků. V následujících ukázkách jsou příklady použití metod a vliv zašuměných hran na vizualizaci. Pro vizualizaci povrchu byla použita funkce *meshz*, která na okrajích zobrazeného povrchu použije funkci *waterfall*. Nalezený povrchu části objektu, který byl snímán, je tedy zobrazen tak, jako by část byla vyjmuta ze svého okolí.



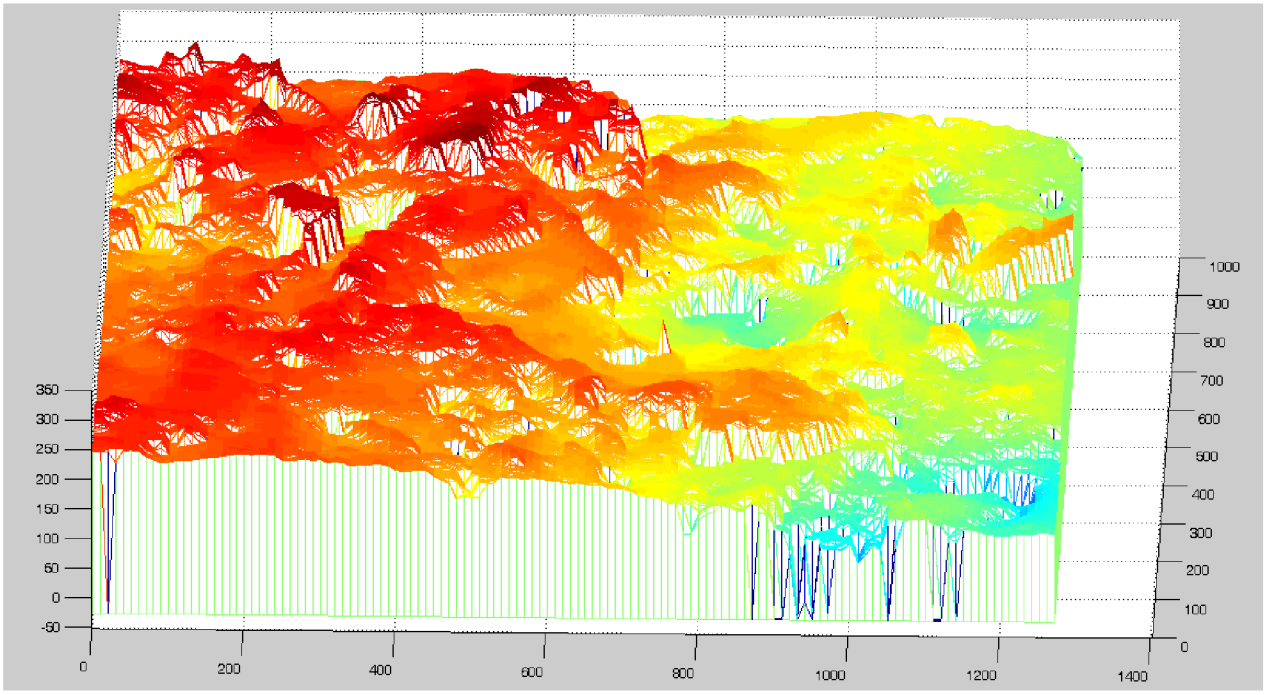
Obrázek 6.8: Zobrazení povrchu pomocí funkce *plot3*



Obrázek 6.9: Zobrazení povrchu pomocí funkce *contour*



Obrázek 6.10: Zobrazení povrchu pomocí funkce *mesh*



Obrázek 6.11: Zobrazení povrchu pomocí funkce *meshz* (použita jiná vstupní data než v předchozích příkladech)

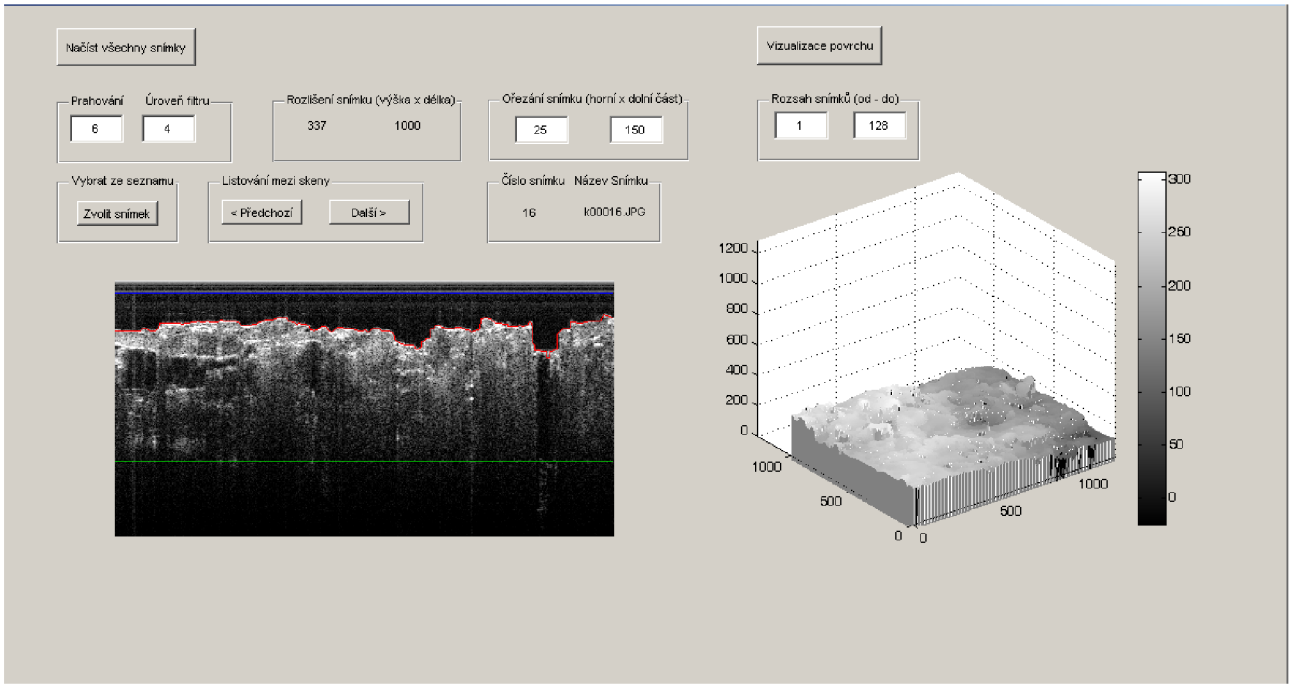
U všech metod byla použita jako detektor hrany funkce „Padající kapky“. U vizualizace pomocí funkce *contour* nejsou píky (tedy neodstraněný artefakt ve vertikálním směru, vyhodnocen jako hrana, případně špatně vyhodnocena informace o chybějící části hrany ve snímku) téměř viditelné. U ostatních metod tyto artefakty výrazně znehodnocují informaci o zobrazeném povrchu. Z tohoto důvodu je nutné detekovat hrany co nejspolehlivěji.

7 Návod k ovládání programu

V této kapitole bude představeno uživatelské prostředí programu (dále jen jako GUI). Jedná se o GUI v programovacím jazyce Matlab. GUI obsahuje funkce pro detekci hrany a vizualizaci povrchu, editovatelné okna pro změnu některých parametrů a prostor pro zobrazení detekované hrany v aktuálním snímku a zobrazení modelu povrchu. Modelem lze libovolně otáčet pomocí držení levého tlačítka a pohybem myši nad modelem. Na obrázku 7.1 je ukázáno GUI a níže jsou popsány jeho jednotlivé části a rozsahy, ve kterých by se měla vstupní data případně upravovat, aby nebyla narušena funkce programu.

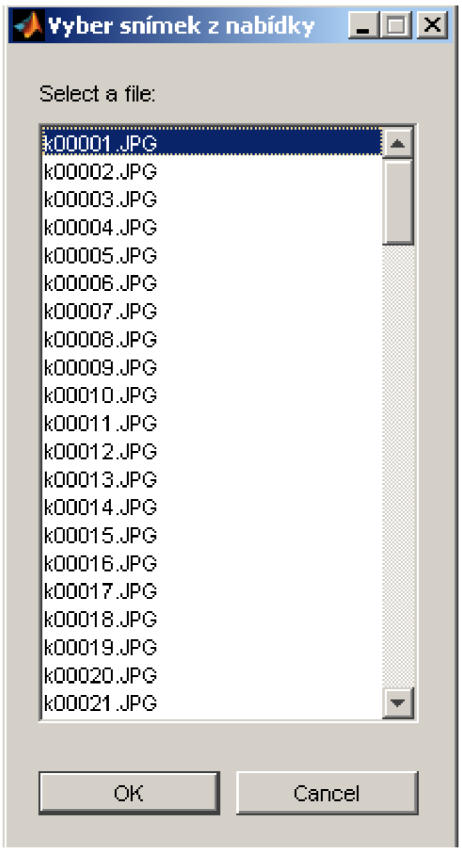
7.1 Popis Uživatelského prostředí

Hlavní soubory se jmenují *final_v1.fig* a *final_v1.m*. Zatímco v prvním souboru je uloženo samotné grafické prostředí, v druhém je uložena hlavní část zdrojové kódu. Pro správný chod programu je nutné mít ve složce s hlavními soubory i soubor s další částí zdrojového kódu *kan.m* a složku Data, ve které jsou uloženy jednotlivé B-scany ve formátu JPG.



Obrázek 7.1: Uživatelské prostředí GUI

Obrázek 7.2: Ukazatel počtu zpracovaných snímků



Obrázek 7.3: Ukázka okna, které se zobrazí po kliknutí na tlačítko *Zvolit snímek*

GUI je rozděleno na dvě hlavní části. Levá část je zaměřena na detekci hran a pravá na vizualizaci povrchu nasnímaného objektu.

Tlačítko *Načíst všechny snímky* slouží k načtení snímků ve formátu *JPG*. Tyto snímky jsou načteny ze složky *Data* v adresáři s programem. Cestu k adresáři s uloženými snímky lze změnit poměrně jednoduchým zásahem do zdrojového kódu.

Dalšími prvky jsou editovatelná okna *Prahování* a *Úroveň filtru*. *Prahování* je parametrem Cannyho operátoru, který zadává pevně silnější prahovací hodnotu. Slabší práh je dopočítán automaticky ve funkci *edge(Canny)*. Hodnota *Prahování* ani nesmí být záporná, *Úroveň filtru* musí mít hodnotu minimálně 1. Vyšší hodnoty *Prahování* mají za následek menšího počtu detekovaných hran, nižší hodnoty zvyšují podíl detekovaného šumu jako hrany. *Úroveň filtru* je parametr mediánového filtru, který má za úkol odstranit vertikální píky. Nižší hodnoty snižují počet odstraněných artefaktů, vyšší hodnoty odstraňují některé detaily ve snímku. Hodnota úrovně filtru nesmí být záporná ani rovna nule.

Rozlišení snímku ukazuje rozlišení původního snímku v pixelech.

Ořezání snímku slouží k odstranění horní či dolní části snímku. Zadané hodnoty určují, jak vysoký pás pixelů bude odstraněn. Právě v horní části snímku se objevují nejsilnější artefakty a zároveň nenese žádnou informaci o povrchu snímaného objektu. Ořezáním této části tedy efektivně zbavíme snímek části šumu, aniž by to mělo dopad na kvalitu snímku. Dolní část také nenese žádnou důležitou informaci o povrchu objektu. Odstraněním této části urychlíme proces detekce hrany i samotnou vizualizaci povrchu. Ani jedna z hodnot nesmí být záporná a uživatel by měl dávat pozor, aby nezaměnil tyto hodnoty. Doporučuje se části odstraňovat postupně, hranice horní (a dolní) ořezané části je zobrazena na snímku modrou (a zelenou) přímkou.

Tlačítko *Zvolit snímek* slouží k otevření seznamu načtených snímků a libovolný výběr snímku. Tento snímek se poté objeví i s detekovanou hranou v GUI. Tlačítka *Předchozí* a *Další* slouží k postupnému procházení mezi snímky oběma směry. *Číslo snímku* a *Název snímku* nese informaci o pořadí snímku a jeho názvu. Tlačítko *Vizualizace* slouží ke spuštění algoritmu, který po detekování hran ve všech načtených snímcích vytvoří pomocí funkce *meshz* povrch nasnímaného objektu. *Rozsah snímků* slouží k výběru, od kterého - po který z načtených snímků se má povrch objektu vypočítat a vykreslit.

7.2 Doporučený postup

V této části práci bude sepsán postup, který by měl uživatel následovat. Po spuštění programu je nutné načíst data. Jak již bylo zmíněno v předchozí kapitole, je nutné, aby byly snímky ve formátu JPG a ve složce Data v adresáři s programem. Stiskem tlačítka *Načíst všechny snímky* dojde k načtení snímků. Poté může uživatel přímo spustit výpočet nutný k zobrazení povrchu za použití defaultního nebo uživatelem upraveného nastavení. Před nebo i po proběhnutí výpočtů Vizualizace a zobrazení povrchu objektu může uživatel měnit parametry, procházet jednotlivé snímky a opětovně spustit funkci vizualizace za použití jiných parametrů. V případě potřeby může uživatel opět stisknout tlačítko *Načíst všechny snímky*, které opět načte snímky a vrátí parametry do defaultního nastavení.

8 Zdrojový kód

V této kapitole se seznámíme s jednotlivými částmi zdrojového kódu. A dílčími výstupy. Části budou seřazeny chronologicky, jak by měly probíhat při běžném používání programu.

8.1 Krok 1. Načtení dat

```
global data                 %označení proměnné data jako globální proměnné
data=dir('Data/*.jpg');          %načtení informací o snímcích do struktury
[s]=1;           %tato proměnná bude používána jako index pro volání snímků
pocetSnimku=length(data);            %zjištění počtu snímků ve složce Data
set(handles.text2,'String',0);   %nastavení vstupních parametrů
%pořadí snímku je 0, protože není zobrazen žádný snímek
set(handles.text3,'String',0);  %název snímku není zobrazen ze stejného důvodu
set(handles.edit6,'string',pocetSnimku);%nastavení maximálního rozsahu snímků
set(handles.edit7,'string',s);
```

8.2 Krok 2. Detekce hrany

```
AktSnimek=data(s,1).name;                    %získání názvu snímku
set(handles.text2,'String',s);               %aktualizace pořadí snímku v GUI
set(handles.text3,'String',AktSnimek);       %aktualizace názvu aktuálního snímku
v GUI
A = imread(AktSnimek);                       %načtení snímku
A=rgb2gray(A);                               %převod z formátu RGB na šedý obraz
[v,d]=size(A);                               %rozlišení snímku
obr=A;                                       %záloha snímku do proměnné 'obr'
hor=get(handles.edit2,'string');             %získání hranice ořezání horní části
snímku
dol=get(handles.edit3,'string');             %získání hranice ořezání dolní části
snímku
hor=str2double(hor);                                %převod proměnných z formátu
'string' na formát 'double'
dol=str2double(dol);
dol=v-dol;                                   %přepočet kvůli indexování (ad řádek
124)
set(handles.text4,'String',d);               %
set(handles.text5,'String',dol-hor);         %odeslání rozlišení ořezaného snímku
do GUI
A=A(hor:dol,:);                              %ořezání snímku
prah=get(handles.edit4,'string');            %získání úrovně prahu z GUI
prah=(str2double(prah))/10;                  %převod a přepočet-prahování Cannyho
operátoru je v rozmezí 0-1
B  =  im2uint8(edge(A,'canny',prah));             %aplikace cannyho operátoru na
snímek
B=B(:,2:end-1);                               %odstranění krajních sloupců, které
jsou prázdné
[m,n]=size(B);                               %získání rozlišení snímku
X=1:n;                                       %nastavení první osy
fil=get(handles.edit1,'string');              %získání a převod úrovně filtrace z
GUI
fil=((str2double(fil))*2)-1;                 %přepočet úrovně na liché číslo
[y,g]=kan(B,fil,m,n,dol,hor);                 %funkce, zaznamenávající souřadnice
hran
axes(handles.axes1);                                  %zobrazení původního snímku,
detekovaných hran a označení ořezaných částí obrazu do GUI
```

```
imshow(obr)                                    %původní (neupravený) snímek
hold on
plot(X,y,'r')                                      %zakreslení detekované hrany do
obrazu
hold on
plot(1:n,hor,'b')                               %zakreslení horní (a dolní) hranice
ořezání obrazu
hold on
plot(1:n,dol,'g')
```

Funkce *[y,g]=kan(B,fil,m,n,dol,hor* slouží k uložení souřadnic detekovaných hran.

```
function [y,g]=kan(B,fil,m,n,dol,hor)

for i=1:n                       %tato část skriptu provádí první detekci hrany
    y(i)=m;                     %používá základní metody "padající kapky"
    c=0;                        %jako hrana je detekován pixel,
    for j=m:-1:1                %který má vyšší hodnotu než 0
        if B(j,i)>c
            y(i)=j;
        end
        B(y(i):end,i)=y(i); %všechny pixely pod detekovanou hranou mají
    end                         %stejnou hodnotu jako hrana.
end

B=medfilt2(B,[1 fil]);          %takto upravený snímek je upraven mediánovým filtrem

for i=1:n                       %konečná detekce hrany
    y(i)=m;
    c=0;
    for j=m:-1:1                %konečná detekce hrany
        if B(j,i)>c
            y(i)=j+hor;         %kompenzace ořezání snímku
        end

        if y(i)==m              %v případě, že není detekována hrana,
            if i==1             %je za hranu považován nejspodnější pixel
            y(i)=m;             %daného sloupce,
            else
            y(i)=y(i-1);        %nebo je použita úroveň hrany v předchozím sloupci
            end
        end
        g(i)=m-y(i);            %přepočet souřadnice hrany pro vizualizaci povrchu
    end
end
```

8.3 Krok 3. Procházení snímků

Zobrazení a detekce jiného snímku je možné dosáhnout třemi způsoby. Buď vybrat snímek z nabídky pomocí tlačítka *Zvolit snímek*. Toto tlačítko otevře seznam všech načtených snímků seřazených podle názvu (viz Obrázek 7.3). Dvojklikem, či označením a stisknutím tlačítka OK, se do proměnné *s* uloží index zvoleného obrázku.

```
for i=1:pocetSnimku
name(i)={data(i,1).name};           %uložení názvů snímků do proměnné 'name'
end
```

```
[s,v]   =   listdlg('Name','Vyber   snímek   z   nabídky','PromptString','Select   a
file:',...
                'SelectionMode','single','listsize',[190 300],...
                'ListString',name);  %vytvoření   a   otevření   okna   se   seznamem
            snímků
```

Nebo je možné procházet databází snímků dopředu a zpátky pomocí tlačítek <*Předchozí* a *Další*>.

```
global data
s=get(handles.text2,'string');              %získání pořadí předchozího snímku
pocetSnimku=length(data);
s=str2double(s);                            %převod z formátu 'string' na formát
'double'
if s<pocetSnimku
s=s+1;                                        %nastavení indexu na další snímek.
Současně tento zabraňuje vybrat snímek mimo databázi.
end
```

respektive

```
global data
s=get(handles.text2,'string');
pocetSnimku=length(data);
s=str2double(s);
if s>1                          %nastavení indexu na předchozí snímek
s=s-1;
end
```

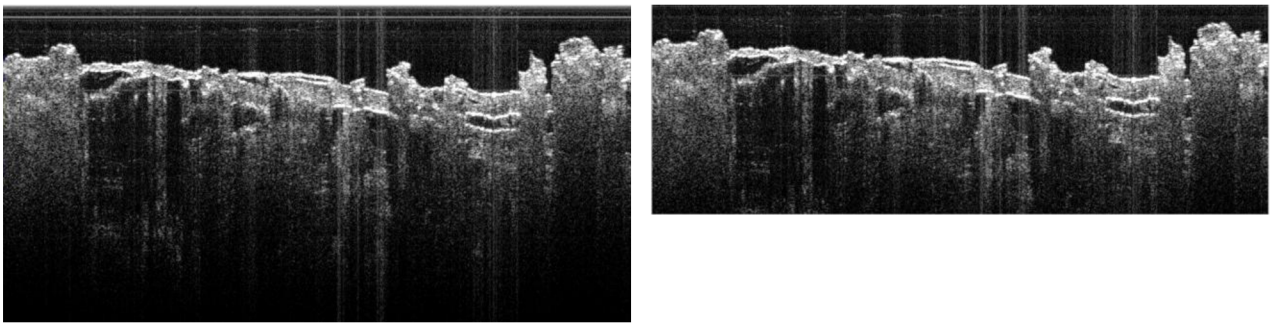
8.4 Krok 4. Vizualizace detekovaného povrchu

```
global data

pocetSnimku=length(data);

h=waitbar(0,'Detekce hran');                %vytvoření okna o stavu procesu
od=get(handles.edit7,'string');              %získání parametrů o rozsahu snímků
z GUI
do=get(handles.edit6,'string');
od=str2double(od);
do=str2double(do);
redSnimku=do-od;
for j=od:do                                 %rozsah snímků
    AktSnimek=data(j,1).name;
waitbar((j-od+1)/(redSnimku+1),h,(['Detekce   hran   ('   num2str(j-od+1)   '/'
num2str(redSnimku+1) ')']));
                                            %zobrazení aktuálního stavu
                                            %procesu
A = imread(AktSnimek);
A=rgb2gray(A);
[v,d]=size(A);
```

```
hor=get(handles.edit2,'string');
dol=get(handles.edit3,'string');
hor=str2double(hor);
dol=str2double(dol);
dol=v-dol;
A=A(hor:dol,:);
obr=A;
prah=get(handles.edit4,'string');
prah=(str2double(prah))/10;
B = im2uint8(edge(A,'canny',prah));
B=B(:,2:end-1);
[m,n]=size(B);
X=1:n;
fil=get(handles.edit1,'string');
fil=((str2double(fil))*2)-1;

[y,g]=kan(B,fil,m,n,dol,hor);

Y(:,j)=g; %uložení nalezené hrany do proměnné Y

end

waitbar(1,h,'Vizualizace povrchu')
Z=od:10:(do*10);        %vytvoření vektoru reprezentující jednotlivé snímky
rozsah=pocetSnimku*10;  %max
axes=(handles.axes2);   %odkaz na prvek GUI, kde se vykreslí vytvořený povrch
rotate3d(axes);         %tento příkaz dovoluje otáčet zobrazeným povrchem

meshz(axes,Z,X,Y);      %výpočet a odeslání vytvořeného povrchu do prvku axes2
set(axes, 'xlim', [0 rozsah], 'ylim', [0 rozsah], 'zlim', [0 rozsah]);%
%nastavení rozsahu os
colorbar('Peer',axes,  'location','EastOutside')  %zobrazení  stupnice  na  pravé
straně od axes2
delete(h);      %zavření okna se stavem procesu
```

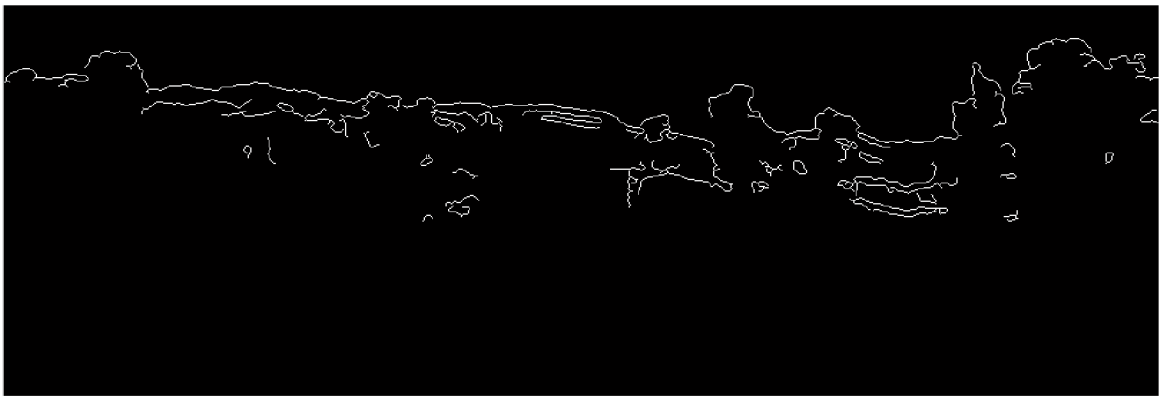
9 Výstupy programu

Tato kapitola je věnována změnám, kterými snímek prochází. Na začátku je vhodné snímek zbavit částí, které nejsou pro detekci povrchu nezbytně nutné. Jak již bylo zmíněno v předchozích kapitolách, horní část snímku je značně postižena artefakty, způsobené optickým systémem OCT přístroje. Dolní část nenese téměř žádnou informaci o snímaném objektu obecně.



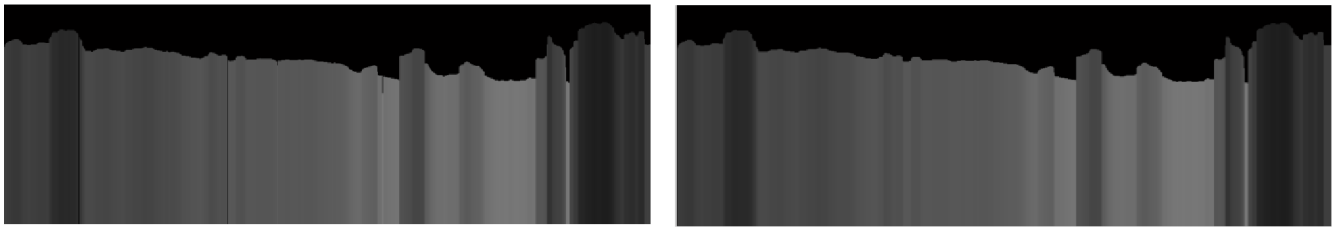
Obrázek 9.1: Původní snímek (vlevo) a ořezaný snímek (vpravo)

V dalších krocích jsou pomocí Cannyho operátoru detekovány hrany.



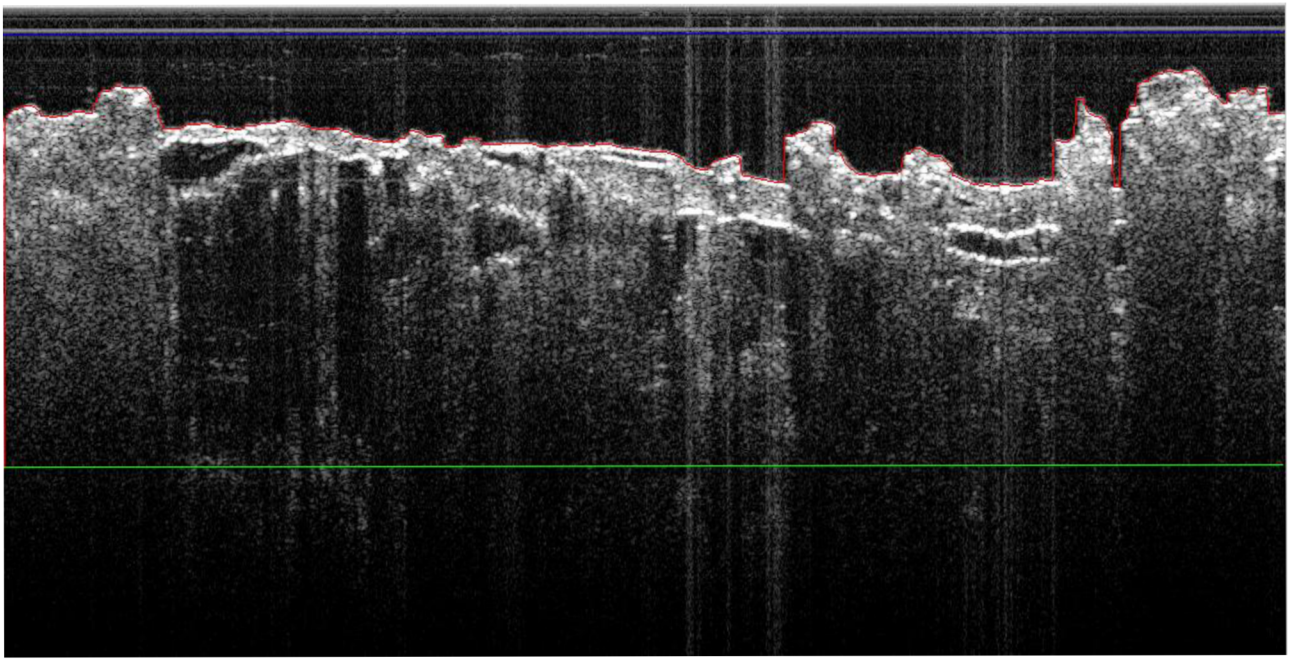
Obrázek 9.2: Zobrazení detekovaných hran

V další části programu je nutné odstranit některé artefakty. Pomocí jednořádkového mediánového filtru o uživatelem zvolené délce lze odstranit vertikální artefakty a úzké úseky nedetekované hrany. Tento přístup ovšem neodstraňuje horizontální artefakty, ale navíc zabraňuje jakémukoliv způsobu tento šum odstranit.

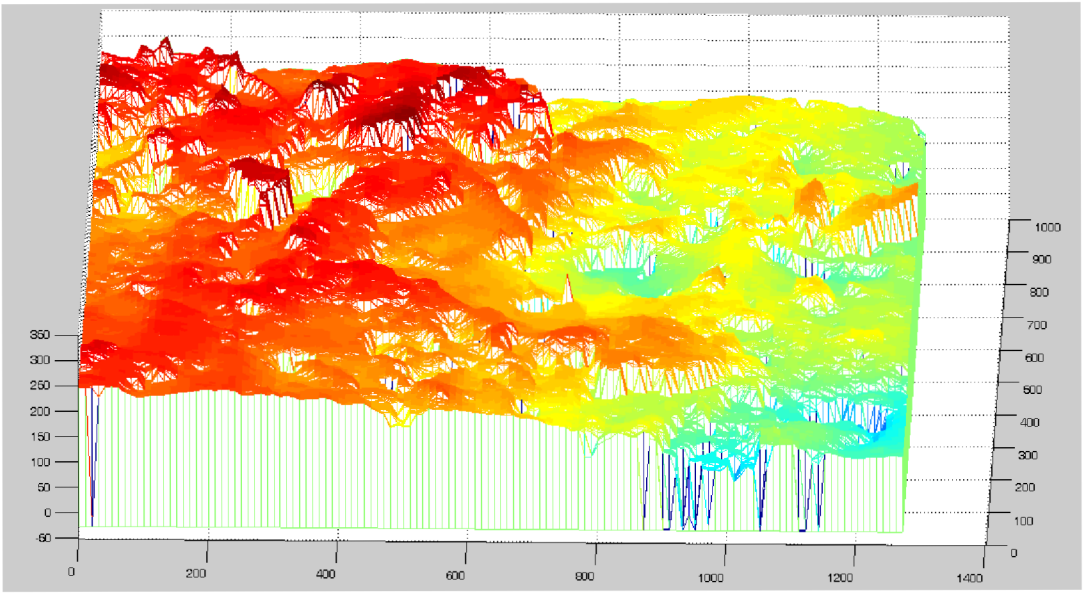


Obrázek 9.3: Odstranění menších nespojitostí v hranách a artefaktů ve vertikálním směru. Vlevo je snímek dále upraven metodou padající kapky, vpravo je snímek upraven mediánovým filtrem.

Na závěr je detekována hrana opět pomocí padající kapky a souřadnice uloženy do matice. Při prohlídce detekovaných hran je v obraze navíc informace o šířce ořezaných částí snímku (viz Obrázek 9.4).



Obrázek 9.4: Původní snímek s detekovanou hranou (červeně), horní hranicí zpracovávaného snímku (modře) a dolní hranicí zpracovávaného snímku (zeleně)



Obrázek 9.5: Zobrazení povrchu pomocí funkce *meshz*

10 Závěr

Cílem této práce bylo provést literární rešerši v oblasti akvizice obrazů OCT, jejich zpracování a analýzy se zaměřením na možnosti nalezení hranice povrchu tvrdé tkáně a jeho vizualizace, vytvoření grafického uživatelského prostředí GUI a jeho otestování na OCT objemových datech.

První kapitoly se zabývají především teoretickému úvodu zpracování obrazu a detekci hran (kapitoly 2-4). Problematice v oblasti akvizice obrazů se věnuje následující kapitola (kapitola 5) Část této kapitoly se zabývá různými módy, ve kterých OCT může pracovat. Šestá kapitola se věnuje práci v prostředí Matlab, je zde návrh sytému pro akvizici a zpracování objemových OCT dat za účelem nalezení povrchu snímaných objektů, srovnání účinnosti testovaných metod jak detekce hrany, tak i metod vizualizace povrchu. Další kapitola je již věnována vytvořenému prostředí GUI. Je zde podrobný popis uživatelského prostředí a jeho funkcemi. Kapitola 8 obsahuje části zdrojového kódu a vysvětlení jejich funkce. Poslední kapitola ukazuje změny, kterým je každý snímek podroben, než je možné vytvořit model povrchu.

Hlavní cíle této práce, tedy seznámit se s optickou koherentní tomografií a vytvoření grafického uživatelského prostředí, které vyhledává hranice povrchu tkáně a zobrazuje ji, byly dosaženy. Přestože nejsou k vyhledávání hranic povrchu ani k její vizualizaci použity nejpokročilejší metody, výsledek přesto do jisté míry věrně kopíruje skutečný povrch objektu. Práce seznamuje také s několika alternativními metodami, které většinou nejsou pro potřeby vizualizace povrchu nejvhodnější, ale mohou nést jinou informaci, která je ve výsledcích jiných metod potlačena.

11 Použitá literatura

[1] APPLE Inc.; Performing Convolution Operations, [online]. iOS Developer Library. 2011, updated 2011-10-12. Dostupný z WWW: [http://developer.apple.com/library/IOs/#documentation/Performance/Conceptual/vImage/ConvolutionOperations/ConvolutionOperations.html].

[2] HLAVÁČ, Václav.; Hledání hran, [online]. FELK ČVUT v Praze, katedra kybernetiky (bez udání data). Citováno dne 30. 12. 2011. Dostupný z WWW: [http://cmp.felk.cvut.cz/~hlavac/Public/TeachingLectures/DetekceHran.pdf].

[3] JANDA, Miloš.; Detekce hran pomocí neuronové sítě, diplomová práce, [online]. Brno, FIT VUT v Brně, 2010. Dostupný z WWW: [http://www.fit.vutbr.cz/study/DP/rpfile.php?id=6583].

[4] KINLAY, Jonathan.; Long Memory and Regime Shifts in Asset Volatility, [online]. Quantitative research and trading. 9. 6. 2011. (Původní vydání: Long Memory and Regime Shifts in Asset Volatility, The best of Wilmott Vol 1 2005). Dostupný z WWW: [http://jonathankinlay.com/index.php/2011/03/long-memory-and-regime-shifts-in-asset-volatility/].

[5] NIXON, Mark S.; AGUADO Alberto S.; Feature and Image Processing. 1st ed. Woburn, MA: Reed educational and Professional Publishing Ltd., 2002. 360 p. ISBN 0-7506 5078-8.

[6] RAYMOND, H. Chan a kol.; An Iterative Procedure for Removing Random-Valued Impulse Noise, [online]., IEEE Signal Proc. Letters, 11 (2004), 921-924. Dostupný z WWW: [http://www.math.cuhk.edu.hk/~rchan/paper/chn/30percentnoise.html].

[7] STRAKA, Stanislav.; Segmentace obrazu, diplomová práce, [online]. Brno, MU FI v Brně. 2009. Dostupný z WWW: [http://is.muni.cz/th/72784/fi_m/dp.pdf].

[8] ŠPAČEK, Radim.; ČERVINKA, Pavel.; Nové možnosti zobrazení koronárních tepen-optická koherentní tomografie, [online]. Cor Vasa Supplementum 1, 2009. Dostupný z WWW: [http://www.e-coretvasa.cz/data/view?id=2626].

[9] THORLABS.; Swept Source Optical Coherence Tomography, [online]. (bez udání data). Dostupný z WWW: [http://www.thorlabs.de/catalogpages/v20/1364.pdf].

[10] THORLABS.; OCS1300SS Swept Source OCT Systém User Guide. [online]. Rev H, 3. 6. 2011. Dostupný z WWW: [http://www.thorlabs.de/Thorcat/18100/18100-D02.pdf].

[11] ZÁLESKÁ, Klára. Význam OCT, vyhodnocení vyšetření. Brno, 2012. 106 s. Diplomová práce, Masarykova univerzita, Lékařská fakulta. Vedoucí diplomové práce MUDr. Eva Žampachová