VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ

FACULTY OF MECHANICAL ENGINEERING

ÚSTAV MATEMATIKY

INSTITUTE OF MATHEMATICS

RESTAURACE PŘEEXPONOVANÉHO DIGITÁLNÍHO OBRAZU

RESTORATION OF OVEREXPOSED DIGITAL IMAGE

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — **KRISTÝNA ZONYGOVÁ**

AUTHOR

VEDOUCÍ PRÁCE — **doc. Mgr. PAVEL RAJMIC, Ph.D.**

SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání bakalářské práce

Ústav:	Ústav matematiky
Studentka:	**Kristýna Zonygová**
Studijní program:	Aplikované vědy v inženýrství
Studijní obor:	Matematické inženýrství
Vedoucí práce:	**doc. Mgr. Pavel Rajmic, Ph.D.**
Akademický rok:	2018/19

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

Restaurace přeexponovaného digitálního obrazu

Stručná charakteristika problematiky úkolu:

V praxi často vzniká situace, kdy část obrazu je tzv. přepálená, neboli dojde k saturaci dostupného číslicového vyjádření. Takové obrázky obsahují, zejdnodušeně řečeno, slité plochy bílé barvy. Existují algoritmy na napravení takového jevu; takové metody se snaží na základě modelu "uhodnout" jaká informace by v postižených místech měla být, a to na základě lokálního okolí.

Cíle bakalářské práce:

Cílem práce by bylo nastudovat, implementovat a porovnat z různých pohledů běžné a modernější metody, které pro tento problém byly vynalezeny. Mezi moderní metody řadíme ty založené na tzv. řídkých reprezentacích signálů (viz literaturu).
Nutným předpokladem je algoritmické myšlení, výhodou je znalost programování, nejlépe v MATLABu.

Seznam doporučené literatury:

ELAD, M. Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing, Springer, 2010.

HRBÁČEK, R., RAJMIC, P., VESELÝ, V., ŠPIŘÍK, J. Řídké reprezentace signálů: Úvod do problematiky. Elektrorevue - Internetový časopis, 2011, roč. 2011, č. 50, s. 1-10. ISSN: 1213- 1539.

URL: http://elektrorevue.cz/cz/download/ridke-reprezentace-signalu--uvod-do-problematiky/

HOU, L., JI, H., SHEN, Z. Recovering Over-/Underexposed Regions in Photographs. SIAM Journal on Imaging Sciences, Vol 6, No 4, p. 2213-2235, 2013.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2018/19

V Brně, dne

L. S.

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

Abstrakt
Tato bakalářská práce se zabývá opravou přeexponovaných šedotónových obrazů na základě řídké reprezentace signálu. Pro převedení obrazu do jiné reprezentace byla použita vlnková transformace. V této reprezentaci bylo hledáno řešení konvexní optimalizační úlohy, která měla požadavky na relaxovanou řídkost signálu a hodnoty pixelů restaurovaného obrazu. Konkrétně byl použit proximální Douglas-Rachford algoritmus využívající dva proximální operátory. Realizace byla provedena v programovém prostředí MATLAB s použitím softwaru Wavelet Toolbox. K hodnocení úspěšnosti metody byl použit poměr PSNR (špičkový odstup signálu k šumu). Navržené řešení bylo testováno na 5 náhodných obrazech a porovnáno s výsledky z editoru fotografií Adobe Photoshop Lightroom CC.

Summary
The Bachelor thesis deals with the recovery of overexposed grayscale images based on sparse signal representation. For image conversion to another representation wavelet transform was used. In this representation, the solution of convex optimization which demand on relax signal sparse and restored pixels values was searched for. In this case, the proximal Douglas-Rachford algorithm was applied which uses two proximal operators. The implementation was carried out in numerical computing enviroment MATLAB using the Wavelet Toolbox software. The PSNR (peak signal-to-noise ratio) was utilized to evaluate success rate of proposed method. The method was tested on 5 random images and compared to results of image manipulation software Adobe Photoshop Lightroom CC.

Klíčová slova
Přeexpozice, řídké reprezentace signálu, konvexní optimalizace, Douglas-Rachford algoritmus, vlnková transformace.

Keywords
Overexposure, sparse signal representation, convex optimization, Douglas-Rachford algorithm, wavelet transform.

ZONYGOVÁ, K.*Restaurace přeexponovaného digitálního obrazu*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2019. 51 s. Vedoucí diplomové práce doc. Mgr. Pavel Rajmic, Ph.D..

Prohlašuji, že jsem bakalářskou práci na téma *Restaurace přeexponovaného digitálního obrazu* vypracovala samostatně pod vedením doc. Mgr. Pavla Rajmice, Ph.D. s použitím uvedené literatury.

Kristýna Zonygová

Ráda bych poděkovala vedoucímu bakalářské práce doc. Mgr. Pavlu Rajmicovi, Ph.D. za odborné vedení, cenné rady a zapůjčenou literaturu.

Kristýna Zonygová

Obsah

Seznam obrázků

1 Úvod

V ideálním případě si pro focení fotografií nejprve vhodně nastavíme parametry fotoaparátu, abychom obraz neměli důvod opravovat. Může se ovšem stát, že nemáme čas fotoaparát nastavit, nebo z jakéhokoliv jiného důvodu vyfotíme fotografii nevhodně a nemáme možnost ji vyfotit lépe. Tato práce si dává za cíl takovou fotografii do určité míry opravit. Práce se zaměřuje na přirozené šedotónové obrazy (případně barevné, které jsou na šedotónové převedeny), jež jsou přeexponovány.

Je známo, že lze obrazové informace v nedourčených systémech vyjádřit pomocí lineární kombinace malého počtu vektorů (v tzv. řídké reprezentaci). Což se s výhodou používá například pro komprimování dat. V práci chceme této vlastnosti využít a najít řídkou reprezentaci obrazu, která by se blížila obrazu originálnímu. Pro přirozené obrazy se ukazuje, že mají velmi řídkou reprezentaci ve vlnkové transformaci. V této reprezentaci bylo hledáno řešení konvexní optimalizační úlohy, která měla požadavky na relaxovanou řídkost signálu a hodnoty pixelů restaurovaného obrazu. Konkrétně byl použit proximální Douglas-Rachford algoritmus využívající dva proximální operátory. Realizace byla provedena v programovém prostředí MATLAB s použitím softwaru Wavelet Toolbox. [1]

V kapitole 2 je popsán princip vzniku fotografie (obrazu), co je přeexponovaný obraz a co jej doprovází. Je zde zmíněno i jak takovému jevu předcházet. V kapitole 3 nalezneme teoretický základ pro práci, konkrétněji normy vektoru, báze prostoru, řídké řešení systému lineárních rovnic a vlnkovou transformaci. V kapitole 4 nalezneme návrh řešení problému a jeho realizaci. Testování navržené metody a srovnání s jinou metodou nalezneme v kapitole 5. K práci náleží program realizovaný v programovém prostředí MATLAB, který po vložení obrazu a nastavení parametrů vypočítá obraz nový.

V práci používáme slovo „přepal“ ve významu místa obrazu, kde je shluk pixelů majících hodnotu 255 (tedy bílé místo). Dále obecně používáme pojem signál, který se ovšem dá snadno zobecnit na námi řešený obraz.

Veškeré obrázky, jež nemají uveden zdroj, jsou mé vlastní nebo byly převzaty z volně šiřitelných zdrojů.

2 Digitální obraz

2.1 Vznik digitální fotografie

Vynález fotografie je datován do první poloviny 19. století. V průběhu století se pak postup vzniku fotografie dále zdokonaloval, až byla v první polovině 20. století do fotografie zakomponována i barva. Počátek a vývoj digitální fotografie nastává okolo 20. století, přestože byl její princip použit již dávno předtím (při skládání ornamentů) v podobě mozaiky. [2]

Analogová fotografie je založena na chemickém principu, kdy světlo procházející objektivem dopadá na fotocitlivou vrstvu a ta je pro vznik fotografie dále chemicky zpracována (vyvolání a ustálení snímku). [2]

Při vzniku **digitální fotografie** dopadá světlo skrze objektiv na senzor, ten je složený ze světlocitlivých buněk a v každé buňce se podle intenzity dopadajícího světla (fotonů) vytváří náboj. Vzhledem k malé velikosti tohoto náboje je potřeba jej v zesilovači podle nastavené ISO hodnoty zesílit. Následně se A/D (analog/digital) převodníkem převedou analogová data na určitou digitální hodnotu (čím větší náboj, tím vyšší hodnota), která reprezentuje jas a představuje jeden bod obrazu (pixel). Obvykle se pro zápis používá 8 bitů, takže jeden pixel může nabývat $2^8 = 256$ odstínů (v tomto případě 0 představuje černou a 255 bílou barvu, hodnoty mezi jsou odstíny šedé). Velikost senzoru, počet jeho buněk a jejich velikost tak zásadně ovlivňují kvalitu obrazu. Čím má senzor více buněk, tím je schopen zachytit větší detaily a lépe je vykreslit (má k dispozici více pixelů, kterými zachycuje předlohu). Pokud máme při stejném počtu buněk větší senzor, můžeme mít větší jednotlivé buňky. Větší buňky jsou na světlo citlivější a dokáží jej pojmout více za současného menšího vzniku šumu než u menších buněk. [3, 4]

Pro zachycení barevných snímků se používá filtr vložený před snímač. Tento filtr se obvykle skládá z mozaiky tří barev modelu RGB (red, green, blue) – červené, zelené a modré – tyto barvy se pravidelně střídají s tím, že podíl zelené je stejně velký jako červené a modré dohromady a to proto, aby se kompenzovala větší citlivost lidského oka na zelenou barvu. Konkrétní barvu pixelu získáme pomocí tzv. Bayerovy interpolace. [3, 4]

2.2 Reprezentace digitálního obrazu

Digitální obraz můžeme popsat jako množinu bodů (pixelů) obsahujících informaci o hodnotě jasu (šedotónový obraz) nebo barvy (barevný obraz), uspořádaných do pravidelné struktury. [3]

Matematicky digitální obraz o velikosti $S_x \times S_y$ pixelů znázorníme jako

$$f : \{0, \ldots, S_x - 1\} \times \{0, \ldots, S_y - 1\} \to R_1 \times R_2 \times \ldots \times R_p,$$

$$p, S_x, S_y \in \mathbb{N},$$

kde prvky (hodnoty) množiny R_n $(n = 1, \ldots, p)$, kterých může množina nabývat, jsou nezáporné. Mohutnost množiny závisí na počtu bitů použitých pro zápis.

Konkrétně pro $p = 1$ dostáváme šedotónový obraz a v případě osmibitového zápisu je $R_1 = \{0, 1, \ldots, 255\}$ množina jasů. V případě $p = 3$ dostáváme barevný obraz obvykle ve složkách RGB, vyjádřený pomocí tří matic o velikosti původního obrazu $S_x \times S_y$, kde

$R_n = \{0, 1, \ldots, 255\}$ $(n = 1, 2, 3)$ je množina odstínů barevného kanálu (červený, zelený, modrý). [5]

2.3 Expozice

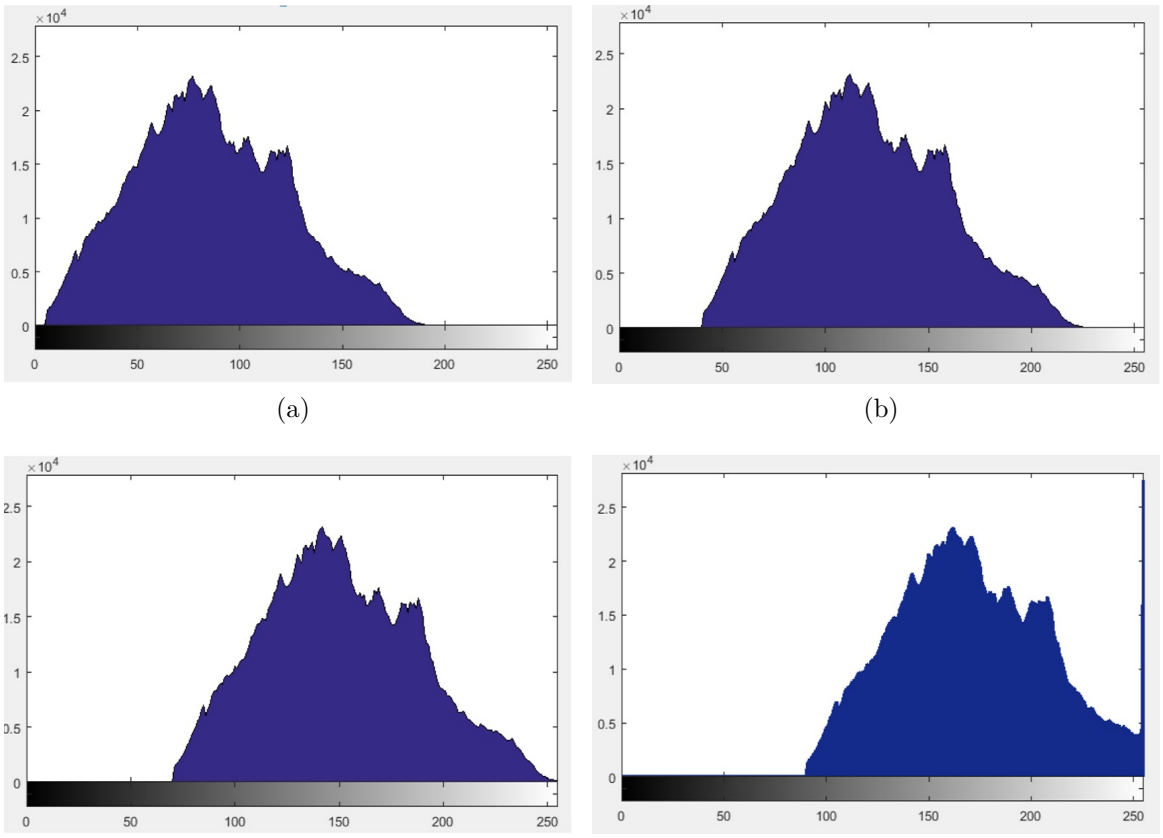
Pro výsledný obraz je významná expozice neboli světlost snímku. Expozici ovlivníme nastavením clony, expozičního času a ISO hodnoty. Clonou nazýváme kruhový otvor, kterým proniká světlo do objektivu při vytváření snímku. Čím větší průměr otvoru nastavíme (menší clonové číslo), tím bude fotka světlejší (otvorem pronikne více světla). Světlost fotografie můžeme ovlivnit také nastavením expozičního času, kdy kratší dobou snímání dostaneme tmavší obraz a naopak při dopadání světla na senzor delší dobu získáme obraz světlejší. Vztah mezi clonou a expozičním časem je následující – chceme-li dvakrát zvětšit clonové číslo (zmenšit otvor, kterým proniká světlo), pak musíme nastavit čtyřikrát delší expoziční čas pro zachování stejné expozice. Nastavením ISO hodnoty elektronicky korigujeme, jak moc je snímač citlivý na světlo (jak moc se zesílí signál). Při nastavování ISO hodnoty musíme být obezřetní, neboť zvyšováním hodnoty ISO zároveň zvyšujeme šum v obraze (nahodilý náboj neodpovídající snímanému obrazu). Ten se projevuje rozostřením hran a nahodilými barevnými pixely viditelnými obzvláště na tmavém pozadí. [3, 6]

K expozici se váží pojmy přeexpozice a podexpozice. Na obrázku 2.1 lze vidět, že přeexponovaný obraz disponuje převahou světlých tónů a velmi málo tmavých. Celkově se jeví příliš světlý. Naopak podexponovaný obraz strádá nedostatkem světla a působí příliš tmavě, což dokazují i jejich histogramy (jejichž princip je vysvětlený v 2.3.3) na obrázku 2.2. U přeexponovaných snímků se často objevují přepaly (obrázek 2.1 (d)). [7, 8]

(a) (b)

(c) (d)

Obrázek 2.1: Expozice obrazu – (a) podexponovaný obraz, (b) výchozí obraz, (c) přeexponovaný obraz, (d) obraz s přepaly



(a) (b)

(c) (d)

Obrázek 2.2: Histogramy odpovídající obrazům z obrázku 2.1 – (a) histogram podexponovaného obrazu, (b) histogram výchozího obraz, (c) histogram přeexponovaného obrazu, (d) histogram obrazu s přepaly

2.3.1 Přepal

Zatímco přeexpozicí rozumíme pouze světlejší obraz, který se dá v případě digitální fotografie poměrně snadno upravit, u přepalu nastává větší problém. Přepal je místo v obraze, které dosáhlo vyšší hodnoty jasu než je snímač schopen při daném nastavení pojmout a zaznamená pouze maximální hodnotu, kterou dokáže zpracovat. Dochází tím ke ztrátě informací o předloze, neboť vzniká bod nebo více bodů obsahujících pouze maximální hodnotu bez další informace o původní kresbě v daném místě. Přepal může vzniknout v jednom nebo více barevných kanálech. Konkrétně v případě šedotónového obrazu se přepal projeví jako největší možná hodnota (255 pro 8bitový zápis) v jeho jediném kanále. [3, 9, 10]

2.3.2 Blooming

Kromě přepalu hrozí další negativní jev a to tzv. blooming, způsobený vlastností snímače. V takovém případě v místech, kde dojde k přepalu (na buňku snímače dopadne více fotonů, než je schopná správně zpracovat), se přebytečné fotony „přelijí“ do dalších buněk a původně nepřepálené pixely jsou saturovány také. [3, 10]

2.3.3 Předcházení přepalu

Jednou z možností předcházení přepalu je **kontrola histogramu**. Histogramem rozumíme graf popisující četnost jasu v obraze. Na jedné ose jsou popsány nebo zobrazeny jasy – obvykle černá nalevo přecházející přes šedé odstíny až napravo po bílou barvu, na ose druhé je popsána jejich četnost. Získáme tak přehled, kolik pixelů v obraze nese daný jas. Histogram můžeme mít také pro každý barevný kanál (obvykle RGB) zvlášť.

Přepal z histogramu poznáme snadno a to tak, že se v histogramu na pravém konci nachází výrazný skok a nejvyšší hodnota (bílá) má mnohem vyšší četnost než okolní jasy (což lze vidět na obrázku 2.2 (d)). Pak je téměř jisté, že k přepalu došlo. Jelikož mnohé fotoaparáty umožňují zobrazovat histogram pořízené fotografie, je vhodné si tento histogram zkontrolovat a případně nastavit parametry fotoaparátu tak, aby k přepalu nedocházelo. Stejně můžeme poznat přepal v dalších kanálech podle jejich samostatných histogramů. [3, 11]

Lze také použít **přechodové filtry**, které se vloží před objektiv tak, aby ztmavily světlé části. Dosáhneme tím snížení dynamického rozsahu scény (rozdíl mezi extrémními jasy bude menší), který už je snímač schopen pojmout. [12]

Další možností je **HDR obraz** (High Dynamic Range). Princip spočívá ve snímání alespoň tří snímků stejné předlohy (lze použít i více), kde pouze měníme expozici, následně z těchto snímků vytvoříme jeden výsledný obraz. Nejprve snímáme normální expozici, poté uděláme snímek podexponovaný (pro dobré vykreslení světlých míst) a přeexponovaný (pro zachování kresby tmavé části). Výsledný obraz z nich složíme tak, že v normálně exponovaném snímku nahradíme tmavé části částmi z přeexponovaného snímku a světlé části částmi z podexponovaného obrazu. Dosáhneme tím zvýšení dynamického rozsahu – tedy že jsou v obraze vykresleny jak dost tmavé části, tak i světlé jasy bez přepalu, čehož bychom při normální expozici kvůli omezení snímače nedosáhli. [13, 14, 12]

3 Teoretická část

3.1 Značení

V této práci bude použito značení, které je uvedeno níže.

Vektory jsou označeny malým tučným písmem (např. $\mathbf{x}, \mathbf{y}$) a pokud nebude uvedeno jinak, je uvažován sloupcový vektor.

Matice jsou značeny velkým tučným písmem (např. $\mathbf{A}, \mathbf{B}$). Jednotlivé prvky matice jsou zapsány příslušným malým písmenem s indexy (např. a_{ij}, b_{ij}). Řádek matice je označen $a_{\bullet j}$ a sloupec matice $a_{i\bullet}$.

Mohutnost množiny – tedy počet prvků množiny – je označen $|\cdot|$.

Otevřený interval je zapsán (a, b), **uzavřený interval** $\langle a, b\rangle$.

Pro označení **násobení matic po složkách** je zaveden operátor $\odot$. Podmínkou je souhlasný rozměr takových matic, pak $\mathbf{A} \odot \mathbf{B} = \mathbf{A}_{ij} \cdot \mathbf{B}_{ij}$.

Pro **vektorizaci matice** (převedení matice na vektor) je zaveden operátor $\mathrm{vec}(\cdot)$, čímž je docíleno seřazení sloupců matice pod sebe do jednoho vektoru, tedy vektorizace matice $\mathbf{A}$ o M řádcích a N sloupcích je zapsána $\mathrm{vec}(\mathbf{A}^{M\times N}) = \begin{pmatrix} a_{i1} \\ a_{i2} \\ \vdots \\ a_{iN} \end{pmatrix}$ pro $i = 1, \ldots, M$.

3.2 l_p norma vektoru

Definice 3.1. l_p norma vektoru $\mathbf{x} \in \mathbb{C}^N$ je definována

$$\|\mathbf{x}\|_p := \left(\sum_{i=1}^{N} |x_i|^p \right)^{\frac{1}{p}} \quad \text{pro } 1 \leq p < \infty, \tag{3.1}$$

speciálně pak

$$\|\mathbf{x}\|_1 := \left(\sum_{i=1}^{N} |x_i| \right) \quad \text{pro } p = 1, \tag{3.2}$$

$$\|\mathbf{x}\|_0 := |\{i : x_i \neq 0, i = 1, \ldots, N\}| \quad \text{pro } p = 0. \tag{3.3}$$

Vzhledem k definici normy se v případě $p = 0$ ve skutečnosti o normu nejedná, přesto tak bude pro jednoduchost značena. [15, 16, 17]

3.3 Frobeniova norma

Definice 3.2. Frobeniova norma matice $\mathbf{A}^{M\times N}$ je definována

$$\|\mathbf{A}\|_F = \|\mathrm{vec}(\mathbf{A})\|_2 = \sqrt{\sum_{i=1}^{M} \sum_{j=1}^{N} |a_{ij}|^2}, \tag{3.4}$$

tedy jako l_2 norma vektorizované matice. [1]

3.4 Báze vektorového prostoru

Bází vektorového prostoru $\mathbb{V}$ konečné dimenze N nazýváme množinu s nejmenším možným počtem lineárně nezávislých vektorů $\{\mathbf{b}_1, \ldots, \mathbf{b}_N\}$, jejichž lineární kombinací dostaneme libovolný vektor $\mathbf{x} \in \mathbb{V}$ daného vektorového prostoru. Tedy

$$\mathbf{x} = \sum_{i=1}^{N} c_i \mathbf{b}_i, \tag{3.5}$$

kde $c_i \in \mathbb{R}$ jednoznačně určují vektor $\mathbf{x}$.

Pro ortogonální bázi $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ platí, že každé její dva vektory jsou na sebe kolmé. Neboli

$$\langle \mathbf{b}_i, \mathbf{b}_j \rangle = 0 \quad \text{pro } i \neq j, \qquad \langle \mathbf{b}_i, \mathbf{b}_i \rangle \neq 0 \tag{3.6}$$

skalární součin dvou různých vektorů báze je roven nule.

Pro ortonormální bázi kromě 3.6 navíc platí, že všechny její prvky $\|\mathbf{b}_i\|_2 = 1$. [18]

3.5 Konvexní funkce

Definice 3.3. Pokud na intervalu I pro každé $a, b \in I$ a každé $\lambda \in \langle 0, 1 \rangle$ platí

$$f(\lambda a + (1 - \lambda)b) \leq \lambda \cdot f(a) + (1 - \lambda) \cdot f(b), \tag{3.7}$$

pak řekneme, že funkce f je na tomto intervalu I konvexní. [19]

3.6 Řídký vektor

Definice 3.4. Vektor $\mathbf{x} \in \mathbb{C}^N$ nazveme k-řídkým, jestliže splňuje $\|\mathbf{x}\|_0 \leq k$.

Tedy vektor řídkosti k má nejvýše k-nenulových složek. [16]

3.7 Řídké řešení systémů lineárních rovnic

Obecně ze všech přípustných řešení $\mathbf{x} \in \mathbb{C}^N$ soustavy lineárních rovnic $\mathbf{A}\mathbf{x} = \mathbf{y}$ vybíráme to, které je nejřidší (tedy obsahuje nejvíce nulových složek vektoru). Matici $\mathbf{A} \in \mathbb{C}^{M \times N}$ nazýváme *slovník*, její sloupce *atomy* a vektor $\mathbf{y} \in \mathbb{C}^M$ představuje naměřená data. Úlohu zapíšeme

$$\min_{\mathbf{x}} \|\mathbf{x}\|_0 \text{ vzhledem k } \mathbf{A}\mathbf{x} = \mathbf{y} \tag{3.8}$$

s předpokladem $M < N$ a plné řádkové hodnosti matice $\mathbf{A}$. [16]

3.7.1 Konvexní optimalizace při zpracování signálů

V mnoha případech řešení problému zpracování signálu může být formulováno jako konvexní optimalizační úloha ve tvaru

$$\arg\min_{\mathbf{x}} f_1(\mathbf{x}) + \ldots + f_m(\mathbf{x}), \tag{3.9}$$

kde $f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}) : \mathbb{R}^N \to (-\infty, \infty)$ jsou konvexní funkce, $\mathbf{x} \in \mathbb{R}$. [20]

3.7.2 l_1 relaxace

Jelikož l_0 norma není konvexní funkce, nelze u problému 3.8 použít konvexní optimalizace. Proto se nabízí zaměnit l_0 normu za l_1 normu, která už konvexní je, neboť řešení obou norem je velmi podobné a ve většině případech dokonce shodné. [16, 17]

3.7.3 Indikátorová funkce

Definice 3.5. Nechť C je neprázdná podmnožina $\mathbb{R}^N$. Pak definujeme

$$\iota_C : \mathbf{x} \mapsto \begin{cases} 0 & \text{pokud } \mathbf{x} \in C \\ \infty & \text{pokud } \mathbf{x} \notin C \end{cases} \tag{3.10}$$

jako indikátorovou funkci množiny C. [20]

3.7.4 Proximální operátor

Definice 3.6. Nechť $f : \mathbb{R}^N \to (-\infty, \infty)$ je zdola polospojitá konvexní funkce splňující $\{\mathbf{x} \in \mathbb{R}^N | f(\mathbf{x}) < \infty\} \neq \emptyset$. Pro každé $\mathbf{x} \in \mathbb{R}^N$ minimalizační problém

$$\underset{\mathbf{y} \in \mathbb{R}^N}{\arg\min}\ f(\mathbf{y}) + \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2 \tag{3.11}$$

zaručuje jednoznačné řešení, které označíme jako $\text{prox}_f \mathbf{x}$. Takto definovaný operátor $\text{prox}_f : \mathbb{R}^N \to \mathbb{R}^N$ nazýváme proximálním operátorem funkce f. [20]

3.7.5 Proximální operátor l_1 normy

Definice 3.7. Pro funkci $f(\mathbf{y}) = \gamma \|\mathbf{y}\|_1$ je proximální operátor tvaru

$$\text{prox}_{\gamma f}(\mathbf{x}) = \underset{\mathbf{y} \in \mathbb{R}^N}{\arg\min}\ \gamma \|\mathbf{y}\|_1 + \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|_2^2 \tag{3.12}$$

což lze rozepsat na

$$\underset{\mathbf{y} \in \mathbb{R}^N}{\arg\min}\ \gamma \sum_{i=1}^{N} |y_i| + \frac{1}{2}\sum_{i=1}^{N} |x_i - y_i|^2. \tag{3.13}$$

Z čehož lze odvodit (blíže popsáno v [21])

$$y_i = \frac{x_i}{|x_i|}\max(|x_i| - \gamma, 0). \tag{3.14}$$

Pak pro $\mathbf{x} \in \mathbb{R}$ měkké prahování s parametrem γ zapíšeme jako

$$\text{soft}_\gamma(\mathbf{x}) := \text{sgn}(\mathbf{x}) \max\left(|\mathbf{x}| - \gamma, 0\right), \tag{3.15}$$

kde $\text{sgn}(\cdot)$ reprezentuje znaménkovou funkci. [22, 23]

3.7.6 Douglas-Rachford algoritmus

Problém 3.9 pro dvě konvexní funkce $f_1(\mathbf{x})$, $f_2(\mathbf{x})$ a libovolné $\gamma \in (0, \infty)$ připouští alespoň jedno řešení, které je charakterizováno dvěma podmínkami

$$\begin{cases} \mathbf{x} = \text{prox}_{\gamma f_2}\mathbf{y} \\ \text{prox}_{\gamma f_2}\mathbf{y} = \text{prox}_{\gamma f_1}\mathbf{y}\,(2\,\text{prox}_{\gamma f_2}\mathbf{y} - \mathbf{y}) \end{cases}$$

Potom můžeme zapsat obecný tvar Douglas-Rachford algoritmu následovně

Algoritmus 3.8 (Obecný Douglas-Rachford algoritmus)**.**
Pro $n = 0, 1, \ldots$

1. $\mathbf{x}_n = \text{prox}_{\gamma f_2}\mathbf{y}_n$,
2. $\lambda_n \in (\varepsilon, 2 - \varepsilon)$,
3. $\mathbf{y}_{n+1} = \mathbf{y}_n + \lambda(\text{prox}_{\gamma f_1}(2\mathbf{x}_n - \mathbf{y}_n) - \mathbf{x}_n)$,

kde zvolíme $\varepsilon \in (0, 1), \gamma > 0$ a $\mathbf{y}_0 \in \mathbb{R}^N$. [20]

3.8 Vlnková transformace

Uvedeme některé transformace, které umožňují převedení signálu do jiné reprezentace. Mezi ně se řadí Fourierova transformace, která zajišťuje převod mezi časovou a frekvenční oblastí a vyjadřuje signál pomocí součtu funkcí sinus a kosinus. Transformaci zapíše

$$F(\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} f(t)\, e^{-i\omega t}\mathrm{d}t, \tag{3.16}$$

kde $f(t)$ je analyzovaný signál. Najdeme ji také v diskrétní podobě jako diskrétní Fourierovu transformaci, ze které dále vychází diskrétní kosinová transformace. Ta rozkládá signál pouze na funkce kosinu a často se využívá pro kompresi dat (např. formát JPEG). Tyto transformace ovšem nejsou schopné lokalizace v čase. Pro převedení do časově frekvenční oblasti lze využít krátkodobou Fourierovu transformaci (vycházející z Fourierovy transformace), jež analyzuje signál postupně po malých částech za využití tzv. časového okna, které je posouváno v čase a frekvencích. Další možností, jak převést signál do časově frekvenční oblasti, je vlnková transformace (ve spojité i diskrétní podobě), která bude použita v navržené metodě a níže bude popsána detailněji. Ta navíc dokáže poskytnout řídkou reprezentaci po částech pravidelných signálů, které mohou obsahovat přechody a singularity. [24]

Vlnková transformace (wavelet transform, WT) patří k takzvaným víceměřítkovým transformacím, které umožňují odděleně zpracovat jak výrazné rysy, tak malé detaily v signálu nebo obraze. Hlavní výhodou oproti jiným metodám (např. Fourierově transformaci) se jeví velmi přesná časová nebo prostorová lokalizace. Může nám podat informace o výskytu frekvencí, nespojitostech signálu nebo vývoji signálu (trendu). S výhodou se tak dá použít pro neperiodické a nestacionární signály (např. pro filtraci nebo kompresi dat bez citelné ztráty informace). Vzhledem k separabilnosti této transformace je možné ji aplikovat i na 2D data (v tomto případě obrazy). [23, 25, 24]

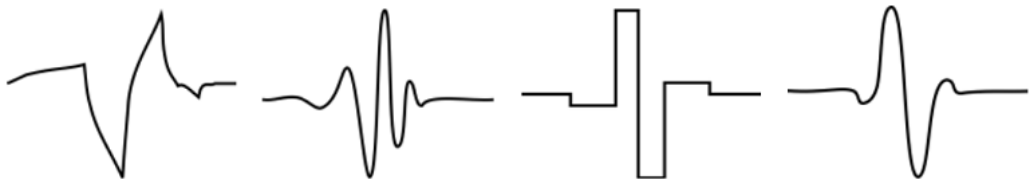
Ortogonální bázi WT (3.4) tvoří časově omezené funkce, které se nazývají vlnky, ty se získají posunutím a změnou měřítka mateřské vlnky. Mateřská vlnka $\psi(t)$ musí splňovat podmínku

$$\int_{-\infty}^{\infty} \psi(t)\mathrm{d}t = 0, \tag{3.17}$$

tedy mít nulovou střední hodnotu (oscilovat). Pro **spojitou vlnkovou transformaci (CWT)** se vlnkové koeficienty $w(s,p)$ vypočítávají podle

$$w(s,p) = \int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{s}}\,\psi^*\left(\frac{t-p}{s}\right)\mathrm{d}t, \tag{3.18}$$

kde $p \in \mathbb{R}$ a $s \in \mathbb{R} - \{0\}$, $f(t)$ je analyzovaný signál, ψ^* komplexně sdružená funkce k mateřské vlnce ψ, p posunutí a s měřítko (scale). Člen $\frac{1}{\sqrt{s}}$ slouží k regulaci energie vlnky při změně měřítka. Posunem je myšleno umístění začátku vlnky na jiné místo časové osy. Při změně měřítka se vlnka buď natahuje (extrahuje) nebo stlačuje (kontrahuje). Čím je větší měřítko, tím vlnka pokrývá větší úsek analyzovaného signálu a podá nám tedy informace o hrubších detailech (nižších frekvencích). Pokud máme vlnku s menším měřítkem (kontrahovanou), můžeme sledovat rychle se měnící detaily (vysoké frekvence). Příklady vlnkových funkcí jsou ukázány na obrázku 3.1. [23, 25, 26, 27]



Obrázek 3.1: Příklady vlnkových funkcí, převzato z [25]

Princip vlnkové transformace:

1. umístíme vlnku s určitým měřítkem do počátku analyzovaného signálu, kde se spočítá vlnkový koeficient (čím je číslo vyšší, tím více je vlnka podobná analyzovanému signálu),
2. vlnka se na časové ose posune doprava a znovu spočítáme koeficient,
3. posouváme vlnku, dokud není pokryt celý signál
4. změníme měřítko vlnky a opět počítáme koeficienty a posouváme vlnku, dokud nevyčerpáme všechna měřítka.

Naproti waveletové transformaci (analýze) za splnění podmínek nulové střední hodnoty a vhodného frekvenčního obsahu mateřské vlnky, existuje také inverzní waveletová transformace (syntéza). Ta umožňuje bezztrátově sestavit původní signál a to s využitím vlnkových koeficientů pro CWT takto:

$$f(t) = \frac{1}{c_\psi}\int_{-\infty}^{\infty}\int_{0}^{\infty} \frac{1}{\sqrt{s}}\psi\left(\frac{t-p}{s}\right) w(s,p)\frac{\mathrm{d}p\,\mathrm{d}s}{s^2}, \tag{3.19}$$

kde $\dfrac{1}{c_\psi}$ je normalizační konstanta. [23, 25, 26, 27]

Pro spojitou vlnkovou transformaci ovšem dostáváme velký objem dat, který může být redundantní. Z tohoto důvodu se využívá **diskrétní vlnková transformace (DWT)**, která se běžně používá i pro obrazy. Zde je použito tzv. dyadické měřítko a posunutí (využívající mocninu dvou), tedy $p = k2^j$ a $s = 2^j$, kde $j, k \in \mathbb{Z}$. Pak pro mateřskou vlnku DWT platí

$$\psi_{jk}(t) = \frac{1}{\sqrt{2^j}} \psi \left(\frac{t - k2^j}{2^j} \right). \tag{3.20}$$

Vlnkové koeficienty DWT počítáme podle

$$w(j,k) = \int_{-\infty}^{\infty} f(t) \frac{1}{\sqrt{s^j}} \psi \left(\frac{t - k2^j}{2^j} \right) \mathrm{d}t. \tag{3.21}$$

Přesně rekonstruovat signál $f(t)$ lze díky ortogonální bázi, pak se inverzní transformace zapíše jako

$$f(t) = k_\psi \sum_{j \in \mathbb{Z}} \sum_{k \in \mathbb{Z}} \psi_{jk}(t) w(j,k), \tag{3.22}$$

kde k_ψ je normalizační konstanta. [23, 26, 27]

Jelikož je možné sestavit ortogonální vlnkovou bázi (kterou budeme využívat) lze DWT zjednodušeně zapsat pomocí maticového zápisu ve tvaru

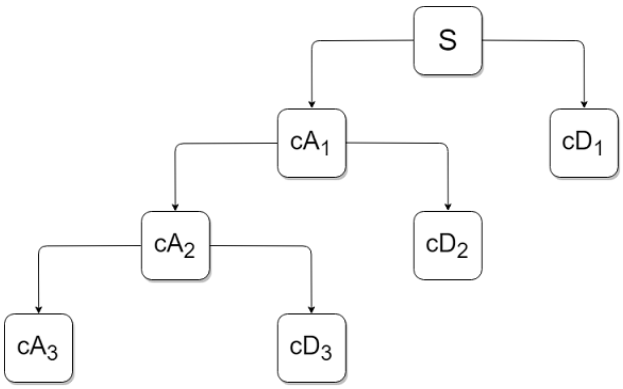
$$\mathbf{c} = \mathbf{M}\mathbf{f}, \tag{3.23}$$

kde $\mathbf{f}$ jsou vstupní data, $\mathbf{c}$ data výstupní a $\mathbf{M}$ transformační matice, jejíž řádky jsou diskretizované ortogonální bázové funkce. Neboť pro ortogonální báze (3.4) platí $\mathbf{M}^{-1} = \mathbf{M}^T$, pak

$$\mathbf{f} = \mathbf{M}^{-1}\mathbf{c} = \mathbf{M}^T\mathbf{c} \tag{3.24}$$

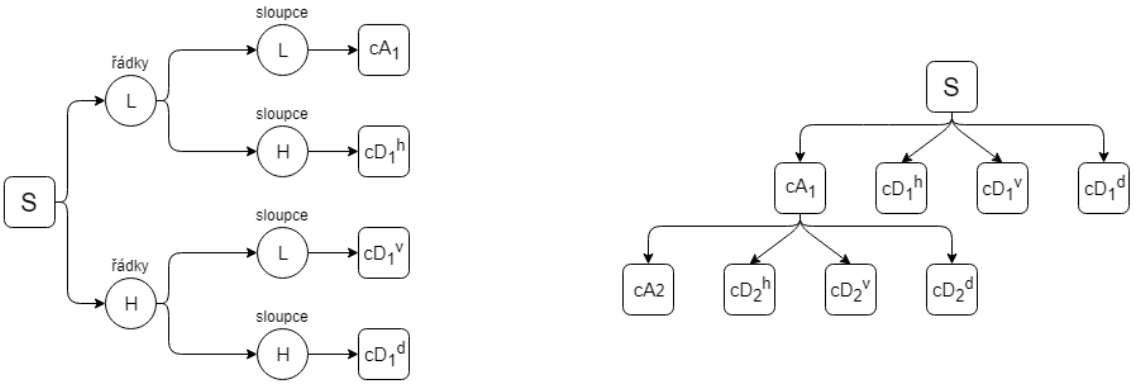
vyjadřuje inverzní diskrétní vlnkovou transformaci (IDWT). [28]

Často používanou realizací diskrétní vlnkové transformace (DWT) je rychlá vlnková transformace (FWT), kdy při dekompozici signálu (analýze) prochází signál dvěma komplementárními filtry (typu horní propust a dolní propust). Tím se signál rozloží na *aproximační vlnkové koeficienty* cA (velká měřítka, nízkofrekvenční obsah) a *detailní vlnkové koeficienty* cD (malá měřítka, vysokofrekvenční obsah). Tato filtrace může být i vícestupňová (tzv. vlnkový dekompoziční strom na obrázku 3.2), kdy se po první filtraci dále stejným způsobem filtrují vždy aproximační koeficienty cA. [23, 25]



Obrázek 3.2: Vlnkový dekompoziční strom 1D. S značí signál, cA_i aproximační vlnkové koeficienty, cD_i detailní vlnkové koeficienty.

Vlnková transformace byla popisována pro jednorozměrný signál, nicméně se dá snadno rozšířit i na obrazy. Díky separabilnosti transformace se nejdříve filtruje obraz po řádcích a následně po sloupcích. Na obrázku 3.3 je naznačen postup dekompozice 2D obrazu pro jednu a dvě úrovně, který je ukázán na příkladě (obrázek 3.4). Rekonstrukce obrazu se provádí obráceným postupem než při dekompozici, kdy všechny podobrazy (vzniklé při dekompozici) prochází filtrací nejdříve po sloupcích, poté po řádcích a výsledné obrazy se sečtou. [23]



Obrázek 3.3: Dekompozice 2D obrazu. Na levé straně dekompozice první úrovně, na pravé straně vlnkový dekompoziční strom obrazu pro úroveň dva. S značí signál, L filtr typu dolní propust, H filtr typu horní propust, cA_i aproximační vlnkové koeficienty, cD_i^h detailní horizontální vlnkové koeficienty, cD_i^v detailní vertikální vlnkové koeficienty, cD_i^d detailní diagonální vlnkové koeficienty. Filtry jsou použity nejprve pro řádky, pak pro sloupce.

Obrázek 3.4: Příklad dekompozice obrazu úrovně 2, převzato z [29]

Dekompozici (analýzu) a následnou rekonstrukci (syntézu) provádíme za účelem úpravy koeficientů, které nám pomohou signál pozměnit podle potřeb (např. odstranění šumu, vyhlazení signálu). Typ filtru a stupeň dekompozice volíme dle charakteru analyzovaných dat. [23, 25]

4 Návrh řešení restaurace obrazu

V této kapitole budeme formulovat úlohu a nastíníme její řešení pomocí řídké reprezentace signálu. Na vstupu budeme mít poškozený obraz $\mathbf{P}$ v podobě matice o velikosti $M \times N$, který obsahuje pixely s maximální hodnotou θ, což je hodnota menší nebo rovna, než která by měla být správně zaznamenána.

4.1 Formulace a řešení problému

Hledáme odhad původních hodnot z poškozeného obrazu $\mathbf{P} \in \mathbb{R}^{M \times N}$, kterých v důsledku chyby nebo nedostatečnosti techniky nenabývá.

Řešení problému budeme hledat jako výsledek konvexní optimalizační úlohy (3.9) tvaru

$$\arg\min_{\mathbf{y}} f_1(\mathbf{y}) + f_2(\mathbf{y}), \tag{4.1}$$

kde $\mathbf{y} \in \mathbb{R}^n$ představuje hledaný restaurovaný obraz v jiné reprezentaci než ve formě obrazu.

Označme matici $\mathbf{M}^r$ (se shodnou velikostí s $\mathbf{P}$) jako matici nesoucí informace o pozicích prvků se správnými hodnotami obrazu $\mathbf{P}$, kde $(\mathbf{M}^r)_{ij} = 1$ reprezentuje správnou hodnotu obrazu $\mathbf{P}$ a $(\mathbf{M}^r)_{ij} = 0$ značí nesprávnou hodnotu obrazu $\mathbf{P}$. Analogicky matice $\mathbf{M}^u$ informuje o pozicích chybných pixelů obrazu $\mathbf{P}$. Pak $\mathbf{M}^r \odot \mathbf{P} = \mathbf{P}^r$ a $\mathbf{M}^u \odot \mathbf{P} = \mathbf{P}^u$, kde $\mathbf{P}^r \in \mathbb{R}^{M \times N}$ obsahuje pouze správné hodnoty a nuly, $\mathbf{P}^u \in \mathbb{R}^{M \times N}$ obsahuje chybné hodnoty obrazu $\mathbf{P}$ a nuly. Písmenem T značíme transformaci – analýzu – vstupních hodnot do jiné reprezentace. Za předpokladu, že T je invertibilní, pak T^{-1} představuje inverzní transformaci – syntézu. Jak již bylo poznamenáno, $\theta \in \mathbb{R}$ představuje maximální hodnotu, která se v obraze nachází.

Vzhledem k předpokladu řídkosti obrazu v určité reprezentaci očekáváme, že nalezneme řídké řešení $\mathbf{y}$ (3.7) a proto úlohu formulujeme jako

$$\arg\min_{\mathbf{y}} \|\mathbf{y}\|_0 \text{ vzhledem k } \begin{cases} \mathbf{M}^r \odot (T^{-1}\mathbf{y}) = \mathbf{P}^r \\ \mathbf{M}^u \odot (T^{-1}\mathbf{y}) \geq \theta \end{cases}, \tag{4.2}$$

kde $(T^{-1}\mathbf{y})$ je syntetizovaný obraz. Tuto úlohu ovšem nelze řešit pomocí konvexní optimalizace (4.1), neboť l_0 norma není konvexní funkce. [17]

Nicméně ve většině případů nám stejné nebo blízké řešení dokáže podat i l_1 norma, která už konvexní funkcí je. Volíme tedy l_1 relaxaci (3.7.2) a úlohu přepíšeme do podoby

$$\arg\min_{\mathbf{y}} \|\mathbf{y}\|_1 \text{ vzhledem k } \begin{cases} \mathbf{M}^r \odot (T^{-1}\mathbf{y}) = \mathbf{P}^r \\ \mathbf{M}^u \odot (T^{-1}\mathbf{y}) \geq \theta \end{cases}. \tag{4.3}$$

Ovšem v tomto tvaru se jedná o úlohu omezenou.

Je tedy ještě potřeba úlohu převést na neomezený tvar a k tomu nám pomůže indikátorová funkce (3.7.3). Neomezenou úlohu zapíšeme ve tvaru

$$\arg\min_{\mathbf{y}} \|\mathbf{y}\|_1 + \iota\{\mathbf{y} : \mathbf{M}^r \odot (T^{-1}\mathbf{y}) = \mathbf{P}^r \ \wedge \ \mathbf{M}^u \odot (T^{-1}\mathbf{y}) \geq \theta\}, \tag{4.4}$$

což lze také zapsat jako

$$\arg\min_{\mathbf{y}} \|\mathbf{y}\|_1 + \iota\{\mathbf{y} : \mathbf{B}_L \leq T^{-1}\mathbf{y} \leq \mathbf{B}_H\}, \text{ kde } \begin{matrix} \mathbf{B}_L = \begin{cases} (\mathbf{P}^r)_{ij} \text{ pro } (\mathbf{M}^r)_{ij} = 1 \\ \theta \text{ pro } (\mathbf{M}^r)_{ij} = 0 \end{cases} \\ \mathbf{B}_H = \begin{cases} (\mathbf{P}^r)_{ij} \text{ pro } (\mathbf{M}^r)_{ij} = 1 \\ \infty \text{ pro } (\mathbf{M}^r)_{ij} = 0 \end{cases} \end{matrix}. \tag{4.5}$$

Jelikož ani jedna z funkcí $f_1(\mathbf{y}) = \|\mathbf{y}\|_1$, $f_2(\mathbf{y}) = \iota\{\mathbf{y} : \mathbf{B}_L \leq T^{-1}\mathbf{y} \leq \mathbf{B}_H\}$ není diferencovatelná, pro další postup zvolíme Douglas-Rachford algoritmus (3.7.6), který sice používá dva proximální operátory (3.7.4), ale nevyžaduje diferencovatelné funkce.

Nejprve si obecně vyjádříme proximální operátor první funkce, kde $\mathbf{x}, \mathbf{z} \in \mathbb{R}^n$

$$\text{prox}_{\gamma f_1}(\mathbf{x}) = \arg\min_{\mathbf{z}} \frac{1}{2}\|\mathbf{x} - \mathbf{z}\|_2^2 + \gamma\|\mathbf{z}\|_1 = \text{soft}_\gamma(\mathbf{x}), \tag{4.6}$$

jedná se o měkké prahování (3.7.5) s prahem γ. [1]

Pro naši úlohu tedy z obecného tvaru Douglas-Rachford algoritmu (3.7.6) dostáváme proximální operátor ($\mathbf{y} \in \mathbb{R}^n$)

$$\begin{aligned} \text{prox}_{\gamma f_1}(2\mathbf{x}_n - \mathbf{y}_n) &= \arg\min_{\mathbf{z}} \frac{1}{2}\|(2\mathbf{x}_n - \mathbf{y}_n) - \mathbf{z}\|_2^2 + \gamma\|\mathbf{z}\|_1 \\ &= \text{soft}_\gamma(2\mathbf{x}_n - \mathbf{y}_n). \end{aligned} \tag{4.7}$$

Označme množinu $C := \{\mathbf{z} : \mathbf{B}_L \leq T^{-1}\mathbf{z} \leq \mathbf{B}_H\}$ a množinu $D := \{\mathbf{Z} : \mathbf{B}_L \leq \mathbf{Z} \leq \mathbf{B}_H\}$, pak proximální operátor druhé funkce je tvaru

$$\begin{aligned} \text{prox}_{\gamma f_2}(\mathbf{y}_n) &= \arg\min_{\mathbf{z}} \frac{1}{2}\|\mathbf{y}_n - \mathbf{z}\|_2^2 + \gamma\, \iota_C \\ &= \text{proj}_C(\mathbf{y}_n) = T\,\text{proj}_D(T^{-1}\mathbf{y}_n) \\ &= T\min(\mathbf{B}_H, \max(T^{-1}\mathbf{y}_n, \mathbf{B}_L)). \end{aligned} \tag{4.8}$$

Jedná se o projekci po složkách na množinu, kde min a max se určuje pro příslušné prvky matice zvlášť. Vzhledem k vlastnostem indikátorové funkce můžeme γ vypustit. [20]

Konkrétní podoba Douglas-Rachford algoritmu (3.7.6) pro naši úlohu je popsána níže. Pro algoritmus jsme zvolili pevné $\lambda = 1$ a diskrétní vlnkovou transformaci T (3.8). Přičemž $\mathbf{y}_{n+1}$ každou iterací konverguje k řešení problému. K ukončení algoritmu dojde po dosažení maximálního počtu iterací nebo po dosažení požadované tolerance relativního rozdílu norem obrazů mezi dvěma iteracemi.

Algoritmus 4.1 (Aplikovaný Douglas-Rachford algoritmus)**.**
Zvolíme $\lambda = 1$, $\gamma > 0$, $\mathbf{y}_0 = T\,\mathbf{P}$.
Pro $n = 1, \ldots,$ MaxIt

1. $\mathbf{x}_n = T\,\text{prox}_{\gamma f_2}(\mathbf{y}_n) = T\,\min(\mathbf{B}_H, \max(T^{-1}\mathbf{y}_n, \mathbf{B}_L))$,
2. $\mathbf{y}_{n+1} = \mathbf{y}_n + \lambda(\text{prox}_{\gamma f_1}(2\mathbf{x}_n - \mathbf{y}_n) - \mathbf{x}_n) = \mathbf{y}_n + \lambda(\text{soft}_\gamma(2\mathbf{x}_n - \mathbf{y}_n) - \mathbf{x}_n)$,

kde MaxIt označuje zadaný maximální počet iterací. Výpočet $\min(\mathbf{B}_H, \max(T^{-1}\mathbf{y}_n, \mathbf{B}_L))$ obrazů $\mathbf{B}_H, T^{-1}\mathbf{y}_n, \mathbf{B}_L$ se provádí po složkách.

Každou iterací tedy získáváme obraz, který má správné hodnoty pixelů stejné jako původní obraz a na pozicích s původně chybnými hodnotami se doplní hodnota θ či více.

4.2 Aplikace v programovém prostředí MATLAB

Realizaci řešení daného problému provádíme v programovém prostředí MATLAB s využitím Wavelet Toolboxu, který poskytuje funkce k analýze a syntéze obrazů.

Pro ověření funkčnosti a spolehlivosti algoritmu si z originálního obrazu vytvoříme obraz poškozený `PicKlip`, který po průchodu algoritmem porovnáváme s původním nepoškozeným obrazem `Pic`. Volbou parametrů se snažíme algoritmus nastavit tak, abychom se restaurovaným obrazem `PicKlip` co nejvíce přiblížili originálnímu obrazu `Pic`.

Algoritmus spouštíme skriptem `startAlg.m` (obrázek 4.1):

- 2. řádek: uložení názvu obrázku do `vstup`
- 3. a 4. řádek: nastavení hodnot `prah` a `prahH`, které budou vysvětleny později
- 5. řádek: volání funkce `pripPic()` (obrázek 4.2)

Dále si ve skriptu `startAlg.m` nastavíme parametry Douglas-Rachford algoritmu `gama`, `lamb`, maximální počet iterací `MaxIt`, toleranci pro ukončení algoritmu `delta`, typ waveletového filtru `wavelet` a úroveň dekompozice `uroven`.

- 8. - 13. řádek: nastavení parametrů Douglas-Rachford algoritmu
- 14. řádek: volání funkce `Doug_Rach()` (obrázek 4.3)
- 17. řádek: volání funkce `vysledek()` (obrázek 4.5)

startAlg.m

```
1   % Priprava obrazu - nacteni obrazu, poskozeni obrazu, priprava obrazu bL a bH
2   vstup = 'frost.jpg';
3   prah = 200;
4   prahH = 400;
5   [Pic,PicKlip,bH,bL] = pripPic(vstup,prah,prahH);
6
7   % Douglas-Rachford algoritmus
8   gama = 0.6;       % prah gama
9   lamb = 1;         % lambda
10  MaxIt = 50;       % maximalni pocet iteraci
11  delta = 10^(-8);  % tolerance pro ukonceni algoritmu
12  wavelet = 'db5';  % typ waveletoveho filtru
13  uroven = 4;       % uroven dekompozice
14  [y,S,iterace] = Doug_Rach(gama,lamb,MaxIt,delta,wavelet,uroven,PicKlip,bL,bH);
15
16  % Vykresleni vysledku
17  [vyslednyObr] = vysledek(y,S,wavelet,Pic,PicKlip);
```

Obrázek 4.1: Sript `startAlg`

Poškozený obraz `PicKlip` se vytváří ve funkci `pripPic()` nahrazením všech hodnot větších než `prah` právě touto hodnotou. Stejným způsobem jsou vytvořeny obrazy `bL` resp. `bH`, kde jsou všechny hodnoty obrazu $\geq$ `prah` nahrazeny hodnotou `prah` resp. ∞ (jež z důvodu proveditelnosti nahrazujeme dostatečně velkým číslem `prahH`).

Popis funkce `pripPic()`:

- 3. řádek: načtení obrazu `vstup` ze složky příkazem `imread()`, převedení na šedotónový obraz příkazem `rgb2gray()` a uložení ve tvaru matice do proměnné `Pic`
- 5. řádek: poškození původního obrazu `Pic` nastavením maximální hodnoty obrazu na θ
- 6. a 7. řádek: příprava obrazu `bH` resp. `bL`, kde jsou maximální hodnoty obrazu nastaveny na `prah` resp. `prahH`

pripPic.m

```
1  function [Pic,PicKlip,bH,bL] = pripPic(vstup,prah,prahH)
2
3      Pic = rgb2gray(imread(vstup)); % nacteni obrazu a prevod na sedotonovy obraz
4      PicKlip = double(Pic); bH = double(Pic);
5      PicKlip(Pic >= prah) = prah;    % poskozeni obrazu
6      bH(Pic >= prah) = prahH;        % priprava obrazu bH
7      bL = PicKlip;                   % priprava obrazu bL
8
9  end
```

Obrázek 4.2: Funkce `pripPic`

Samotný Douglas-Rachford algoritmus probíhá ve funkci `Doug_Rach()`, kde pro analýzu obrazů používáme diskrétní vlnkovou transformaci (3.8) příkazem `wavedec2()` a k syntéze `waverec2()`. Nejdříve se analyzuje obraz `PicKlip` a uloží do `y`, což se použije jako počáteční vektor. Následuje cyklus, ve kterém se v každé iteraci nejprve syntetizuje vektor `y` na obraz `y_T`, spočítá proximální operátor `proxF2 = min(bH,max(bL,y_T))` a obraz znovu analyzuje na vektor `x`. Minimum a maximum proximálního operátoru `proxF2` se určuje pro každý prvek obrazu zvlášť. Nové `y` se výpočítá pomocí proximálního operátoru `proxF1`, který volá funkci `soft()` (obrázek 4.4) k výpočtu měkkého prahování. Pro ukončení cyklu při dosažení požadované tolerance se také počítá relativní rozdíl Frobeniových norem (3.3) obrazů `norma` a `normaN` mezi dvěma iteracemi.

Popis funkce `Doug_Rach()`:

- 3. řádek: DW (3.8) analýza obrazu `PicKlip` příkazem `wavedec2()`
- 4. řádek: nastavení počátečního vektoru
- 7. - 23. řádek: cyklus výpočtu restaurovaného obrazu `y`
  - 8. řádek: syntéza vektoru `y` na obraz `y_T`
  - 9. řádek: výpočet proximálního operátoru `proxF2` (v podobě obrazů) (4.8)
  - 10. řádek: analýza `proxF2` do vlnkové reprezentace
  - 11. řádek: výpočet Frobeniovy normy (3.3)
  - 14. řádek: výpočet `proxF1` – voláním funkce `soft()` (obrázek 4.4)
  - 15. řádek: výpočet nového `y`
  - 17. -21. řádek: ověření dosažení tolerance

Doug_Rach.m

```
function [x,S,iterace] = Doug_Rach(gama,lamb,MaxIt,delta,wavelet,uroven,PicKlip,bL,bH)

    [C,S] = wavedec2(PicKlip,uroven,wavelet);     % analyza chybneho obrazu
    y = C;                                        % nastaveni pocatecniho vektoru
    norma=norm(PicKlip,'fro');                    % vypocet normy

    for iterace = 1:MaxIt
        y_T = waverec2(y,S,wavelet);              % synteza y
        proxF2 = min(bH,max(bL,y_T));             % proximalni operator druhe funkce
        [x,~] = wavedec2(proxF2,uroven,wavelet); % analyza proxF2
        normaN = norm(proxF2,'fro');

        Q = 2*x-y;
        proxF1 = soft(Q,gama);                    % proximalni operator prvni funkce
        y = y+lamb*(proxF1-x);                    % vypocet noveho y

        if iterace >= 2                           % podminka ukonceni algoritmu
            if (abs((norma-normaN)/norma) < delta)
                break;
            end
        end
        norma=normaN;
    end
end
```

Obrázek 4.3: Funkce `Doug_Rach`

Popis funkce `soft()`:

- 3. - 5. řádek: měkké prahování (3.7.5)

soft.m

```
function output = soft(Q,gama)
    output=zeros(1,length(Q));
    for i=1:length(Q)
        output(i) = Q(i)/abs(Q(i))*max((abs(Q(i))-gama),0);
    end
end
```

Obrázek 4.4: Funkce `soft`

Po ukončení algoritmu provedeme zavoláním funkce `vysledek()` syntézu vypočítaného obrazu `y` a na závěr tento restaurovaný obraz zobrazíme včetně jeho histogramu.

- 2. řádek: syntéza `y` na `vyslednyObr`
- 6. - 11. řádek: vykreslení obrazů a jejich histogramů (originál `Pic`, vypočítaný `y` a chybný `PicKlip`)

vysledek.m

```
1  function [vyslednyObr] = vysledek(y,S,wavelet,Pic,PicKlip)
2      vyslednyObr = waverec2(y,S,wavelet); % synteza vysledneho vektoru
3      vyslednyObr = uint8(vyslednyObr); % prevod do formatu uint8
4
5      figure;
6      subplot(3,2,1); imshow(Pic); title('original');
7      subplot(3,2,2); imhist(Pic); title('histogram original');
8      subplot(3,2,3); imshow(vyslednyObr); title('spocitany');
9      subplot(3,2,4); imhist(vyslednyObr); title('histogram spocitany');
10     subplot(3,2,5); imshow(uint8(PicKlip)); title('pokazeny');
11     subplot(3,2,6); imhist(uint8(PicKlip)); title('histogram pokazeny');
12 end
```

Obrázek 4.5: Funkce `vysledek`

5 Testování navržené metody

Pro hodnocení kvality restaurování obrazu byl zvolen parametr PSNR (peak signal-to--noise ratio neboli špičkový odstup signálu od šumu). Ten je možné počítat díky tomu, že je k dispozici originální obraz (jelikož z něj jsou vytvářeny obrazy poškozené – přepálené). PSNR poměřuje maximální možnou energii signálu a energii šumu s využitím střední kvadratické chyby (MSE) mezi originálním (referenčním) obrazem a obrazem restaurovaným. Vztah pro výpočet PSNR v decibelech je tvaru

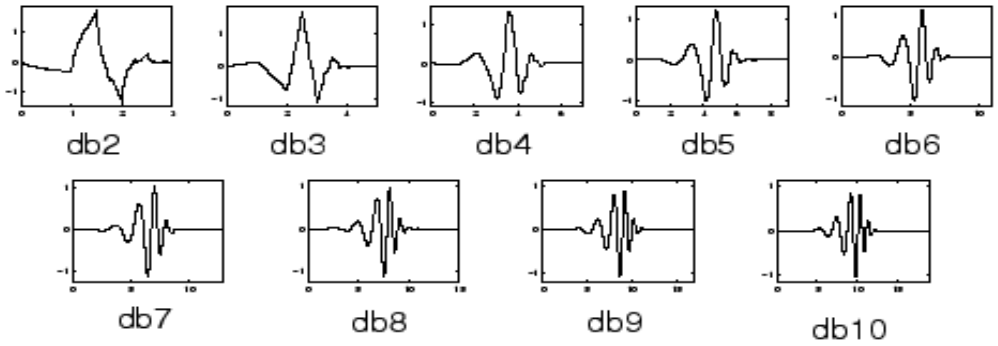
$$\mathrm{PSNR} = 10 \cdot \log_{10} \left(\frac{\mathrm{MAX}_R^2}{\mathrm{MSE}} \right), \tag{5.1}$$

kde střední kvadratická chyba je vyjádřena jako

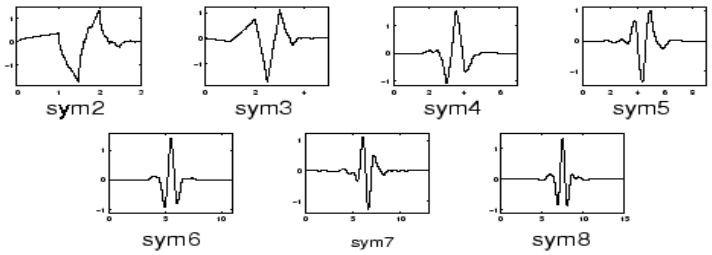
$$\mathrm{MSE} = \frac{1}{M \cdot N} \sum_{i=1}^{M} \sum_{j=1}^{N} \|R_{ij} - P_{ij}\|^2 \tag{5.2}$$

pro referenční obraz R a porovnávaný obraz P o stejné velikosti $M \times N$. Hodnota MAX_R je určena jako největší možná hodnota pixelu referenčního obrazu (pro 8bitové obrazy pak $\mathrm{MAX}_R = 255$). Tedy čím vyšší hodnota PSNR je vypočítána, tím více se porovnávaný obraz blíží referenčnímu. [30, 29]

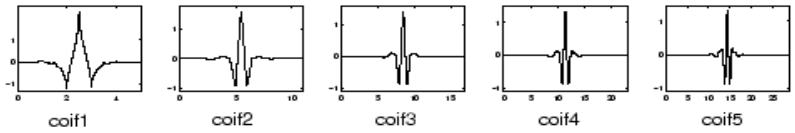
Pro testování algoritmu byly použity ortogonální vlnky obsažené ve WaveletToolboxu (jejich ukázky jsou na obrázku 5.1, 5.2, 5.3). Vlnky typu Daubechies (dbN, N značí řád) jsou často používané a až na nejjednodušší db1 neboli Haarovu vlnku se jedná o spojité, nesymetrické vlnky s filtrem délky 2N, které vykazují fraktálnost. Vlnky typu Symlet (symN, N značí řád) vycházejí z vlnek typu Daubechies a jsou jim velmi podobné. Stejně tak vlnky typu Coiflet (coifN, N značí řád), které mají oproti předchozím filtr délky 6N. [23]



Obrázek 5.1: Vlnky typu Daubechies, převzato z [31]

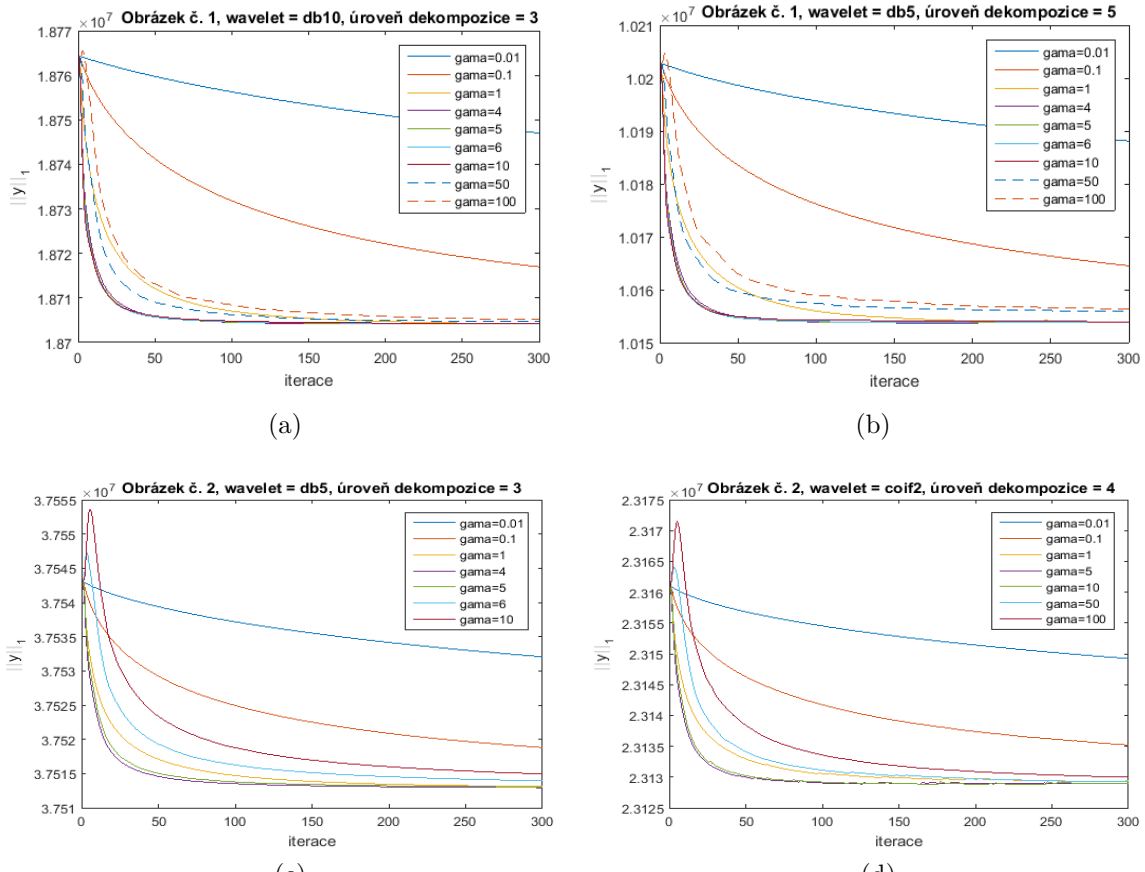


Obrázek 5.2: Vlnky typu Symlets, převzato z [31]



Obrázek 5.3: Vlnky typu Coiflets, převzato z [31]

V grafech na obrázku 5.4 lze vidět, jak se mění l_1 norma obrazu (relaxovaná řídkost) s rostoucím počtem iterací navrženého algoritmu. Hodnoty odpovídají teorii, tedy že pro libovolné $\gamma > 0$ algoritmus konverguje k minimalizaci l_1 normy. Pro ukázku byly vybrány dva různé obrazy (5.5 (a), (b)), různé úrovně dekompozice a typy vlnek. Lze také usoudit, že vhodným zvolením γ dosáhneme rychlejší konvergence k požadovanému řešení.



Obrázek 5.4: Grafy vyjadřující závislost relaxované řídkosti na počtu iterací

Dále bylo náhodně vybráno 5 obrazů (obrázek 5.5), na kterých jsou otestovány některé typy vlnek pro některé úrovně dekompozice na základě hodnoty PSNR. Testování probíhalo pouze pro úroveň dekompozice 2-5 vzhledem k výpočetní náročnosti. Pro ilustraci obraz o velikosti 1198×1920 pro vlnku = db40, úroveň dekompozice = 5, $\delta = 10^{-10}$, $\lambda = 1$, $\gamma = 10$, kde bylo dosaženo 700 iterací a při parametrech počítače – procesor Intel Pentium CPU N3540 (2.16 GHz, 4 jádra), RAM 4GB DDR3 (666 MHz) – algoritmus zpracovával přibližně 54 minut.

Všechny obrazy byly nastaveny na maximální hodnotu pixelu $\theta = 200$, $\lambda = 1$, práh pro měkké prahování $\gamma = 10$, toleranci pro ukončení algoritmu $\delta = 10^{-10}$ a maximální počet iterací MaxIt = 700.

(a) velikost 853×1280 px

(b) velikost 1044×1920 px

(c) velikost 1280×1920 px

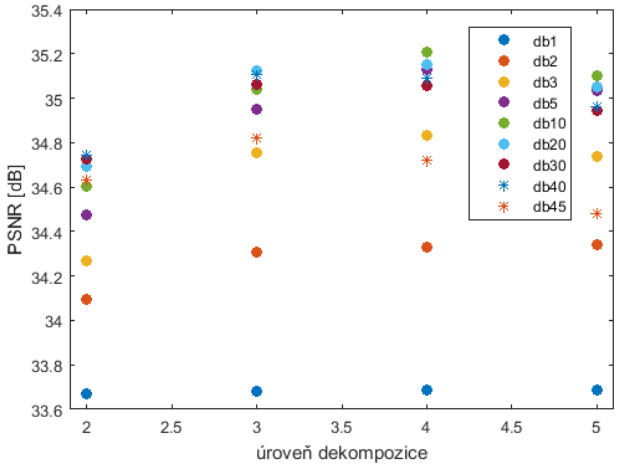
(d) velikost 1280×1920 px

(e) velikost 1198×1920 px

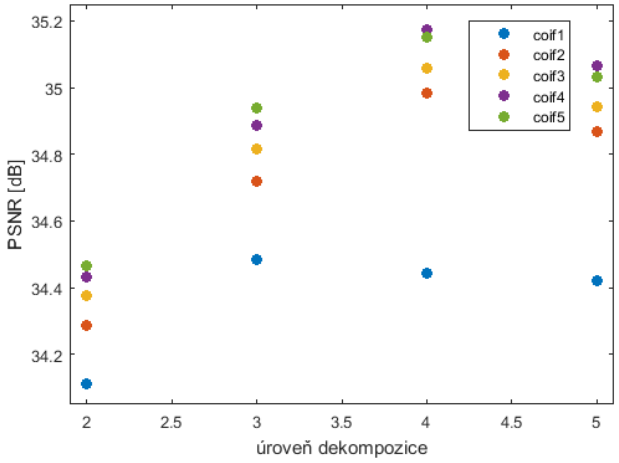
Obrázek 5.5: Obrazy použité pro testování navržené metody včetně jejich velikostí

Výsledky testování lze vidět na obrázcích 5.6, 5.7, 5.8, 5.9, 5.10. Pro lepší čitelnost jsou obrazy také obsahem přílohy.

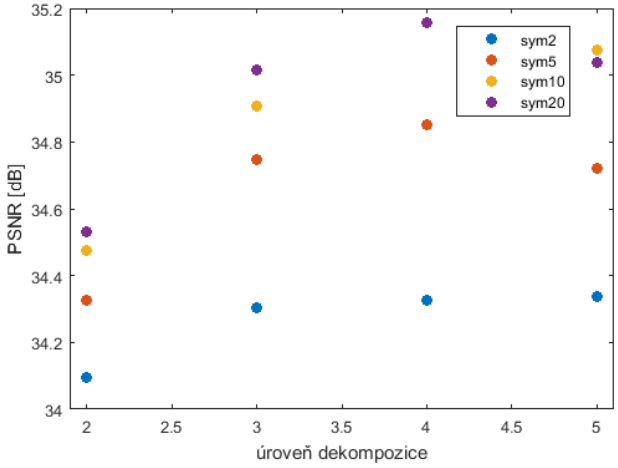
Pro obrázek 5.5 (a) vychází hodnota PSNR nejlépe pro vlnku typu Daubechies řádu 10 (db10) při úrovni dekompozice 4. Velmi podobných výsledků dosahovaly z vlnek typu Daubechies také vlnky db20, db5, db40 a db30. Z vlnek typu Coiflets jsou to coif4 a coif5 také při 4. úrovni dekompozice. Vlnky typu Symlets pak taktéž pro 4. úroveň dekompozice dosahují nejvyššího PSNR pro sym20 a sym10. Nejhorších výsledků dosahují pro všechny testované úrovně dekompozice vlnky nízkého řádu (db1, db2, sym2 a coif1). Hodnot mezi nimi nabývají ostatní vlnky db3, db45, coif2, coif3, sym5. Z výše uvedeného vyplývá, že pro tento obraz vychází nejlépe úroveň dekompozice 4 pro vlnky vyšších řádů, kde příliš nezáleží na typu použité vlnky, jelikož některá vlnka z každého použitého typu dosahuje hodnoty PSNR kolem 35,2.



(a)



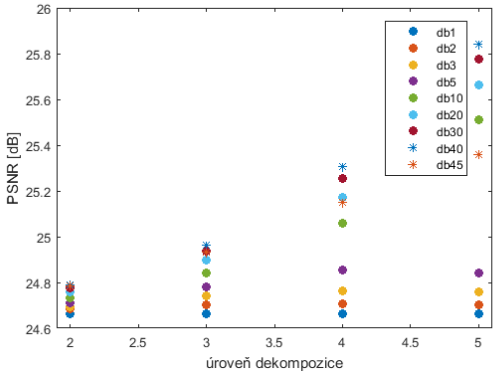
(b)



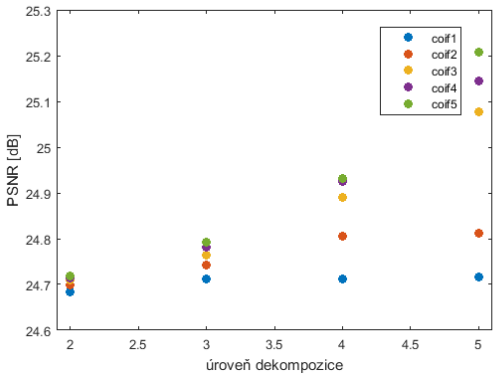
(c)

Obrázek 5.6: Grafy vyjadřující závislost hodnoty PSNR na použitém typu vlnky a úrovně dekompozice pro obraz 5.5 (a)

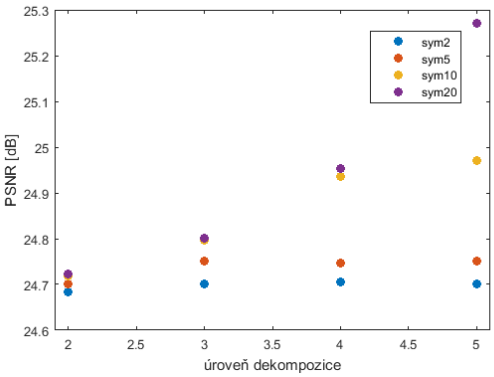
Pro obrázek 5.5 (b) vychází hodnota PSNR=25,8416 nejlépe pro vlnku typu Daubechies řádu 40 (db40) při úrovni dekompozice 5. O několik desetin menší hodnoty PSNR dosahují vlnky coif5 a sym20 pro 5. úroveň dekompozice. Pro všechny typy vlnek vychází nejlépe úroveň dekompozice 5, kde se také projevují největší rozdíly mezi vlnkami nižších řádů (db1, db2, db3, db5, coif1, coif2, sym2, sym5) jež dosahují nízké hodnoty PSNR oproti vyšším řádům (db10, db20, db30, db40, db45, coif3, coif4, coif5, sym10, sym20). Z výše uvedeného vyplývá, že pro tento obraz vychází nejlépe úroveň dekompozice 5 pro vlnky vyšších řádů.



(a)



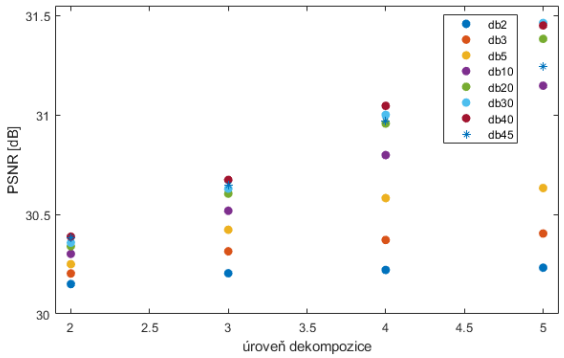
(b)



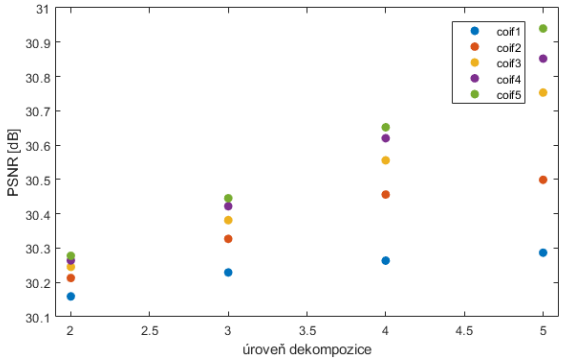
(c)

Obrázek 5.7: Grafy vyjadřující závislost hodnoty PSNR na použitém typu vlnky a úrovně dekompozice pro obraz 5.5 (b)

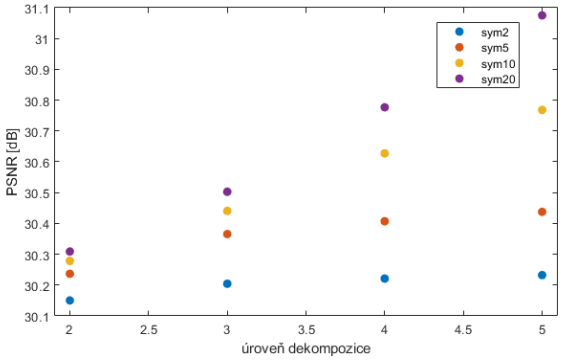
Pro obrázek 5.5 (c) vychází hodnota PSNR=31,4638 nejlépe pro vlnku typu Daubechies řádu 30 (db30) při úrovni dekompozice 5. Velmi podobných výsledků dosahovaly z vlnek typu Daubechies také vlnky db40 a db20. O několik desetin menší hodnoty PSNR dosahují vlnky coif5 a sym20 pro 5. úroveň dekompozice. Pro všechny typy vlnek vychází nejlépe úroveň dekompozice 5. S rostoucí úrovní dekompozice se také projevují větší rozdíly mezi vlnkami nižších řádů (db2, db3, db5, coif1, coif2, sym2, sym5) jež dosahují nízké hodnoty PSNR oproti vyšším řádům (db10, db20, db30, db40, db45, coif3, coif4, coif5, sym10, sym20). Lze tedy říci, že pro tento obraz opět vychází nejlépe úroveň dekompozice 5 pro vlnky vyšších řádů.



(a)



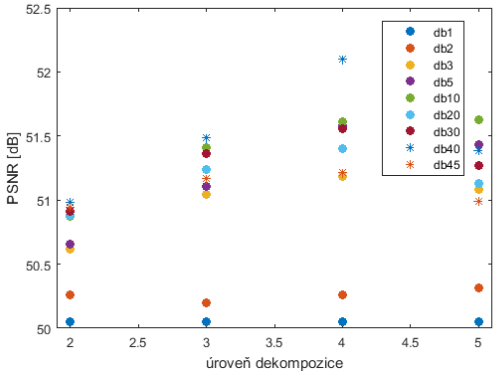
(b)



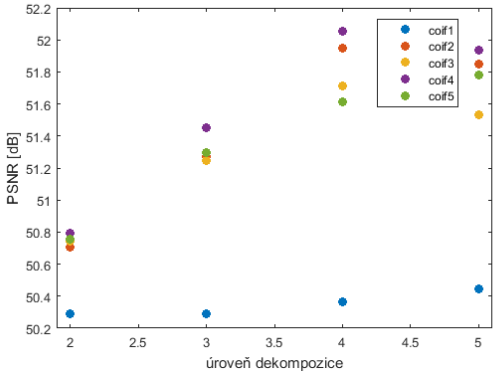
(c)

Obrázek 5.8: Grafy vyjadřující závislost hodnoty PSNR na použitém typu vlnky a úrovně dekompozice pro obraz 5.5 (c)

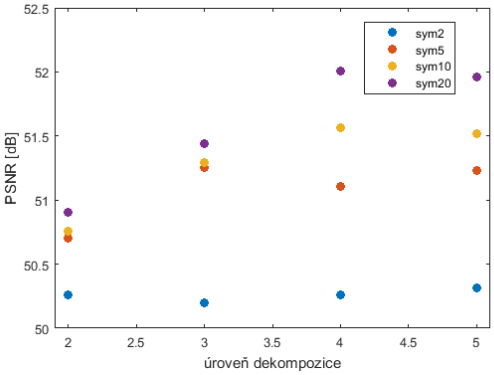
Pro obrázek 5.5 (d) vychází hodnota PSNR=52,1017 nejlépe pro vlnku typu Daubechies řádu 40 (db40) při úrovni dekompozice 4. Velmi podobných výsledků dosahovaly také vlnky vlnky coif4, coif2 a sym20 také pro úroveň dekompozice 4. Nižších hodnot PSNR dosahovaly vlnky řádu menšího než 3, což se projevilo u všech úrovní dekompozice. Pro tento obraz vychází nejlépe úroveň dekompozice 4 opět pro vlnky vyšších řádů, kde příliš nezáleží na použitém typu vlnky, jelikož některá vlnka z každého typu vlnek se blíží nejlepší dosažené hodnotě PSNR.



(a)



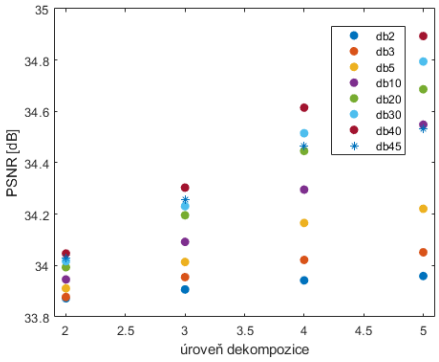
(b)



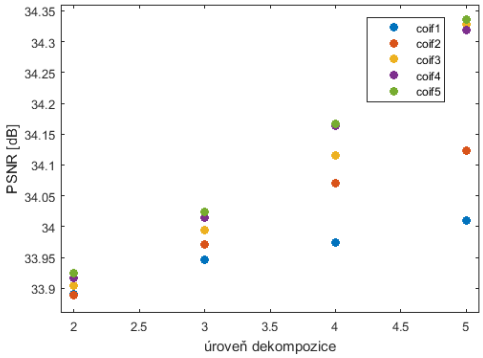
(c)

Obrázek 5.9: Grafy vyjadřující závislost hodnoty PSNR na použitém typu vlnky a úrovně dekompozice pro obraz 5.5 (d)

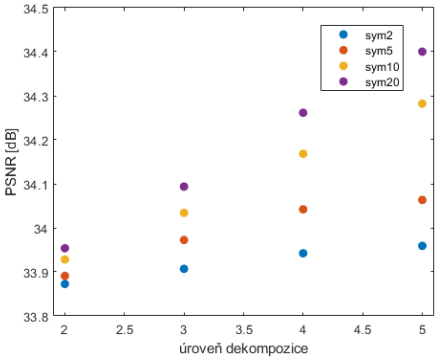
Pro obrázek 5.5 (e) vychází hodnota PSNR=34,8773 nejlépe pro vlnku typu Daubechies řádu 40 (db40) při úrovni dekompozice 5. O několik desetin menší hodnoty PSNR dosahují vlnky db30, coif5, coif3, coif4 a sym20 pro 5. úroveň dekompozice. Pro všechny typy vlnek vychází nejlépe úroveň dekompozice 5. S rostoucí úrovní dekompozice se také projevují větší rozdíly mezi vlnkami nižších řádů (db2, db3, db5, coif1, coif2, sym2, sym5) jež dosahují nízké hodnoty PSNR oproti vyšším řádům (db10, db20, db30, db40, db45, coif3, coif4, coif5, sym10, sym20). Z výše uvedeného vyplývá, že pro tento obraz opět vychází nejlépe úroveň dekompozice 5 pro vlnky vyšších řádů, navíc se výsledky podobají výsledkům pro obraz 5.5 (c).



(a)



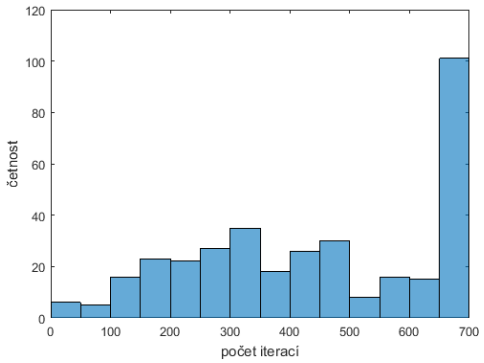
(b)



(c)

Obrázek 5.10: Grafy vyjadřující závislost hodnoty PSNR na použitém typu vlnky a úrovně dekompozice pro obraz 5.5 (e)

Na obrázku 5.11 se nachází graf, který vyhodnocuje kolikrát algoritmus dosáhl určitého počtu iterací pro testované parametry.



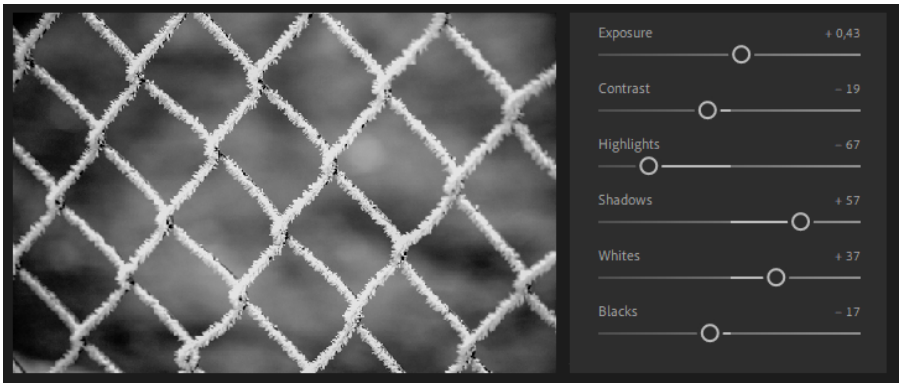
Obrázek 5.11: Histogram vyhodnocující počet iterací algoritmu při parametrech testovaných v této práci

Z grafu na obrázku 5.11 vyplývá, že algoritmus v poměrně velkém množství případů (zhruba 100) dosáhl maximálního počtu iterací. Důvodem mohl být (v kombinaci s ostatními parametry) nepříliš vhodně zvolený parametr γ, v jehož důsledku obraz konvergovat k řídkému vyjádření pomalu. Lze tedy předpokládat, že pro tyto obrazy by při větším počtu iterací bylo možné dosáhnout lepších výsledků.

5.1 Porovnání výsledků navržené metody s editorem fotografií

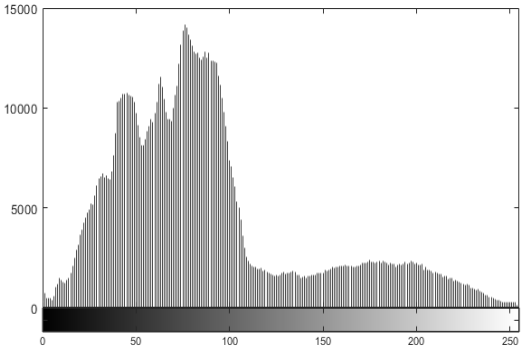
Pro porovnání s běžnými metodami byl využit často používaný editor fotografií Adobe Photoshop Lightroom CC od společnosti Adobe Systems. V něm byla pro větší objektivitu zvolena autokorekce a to vzhledem k tomu, že pro každý obraz vychází lépe jiný typ úpravy (ukázka autokorekce na obrázku 5.12).

Nyní je uveden obraz i jeho histogram pro nejlepší hodnotu PSNR, které bylo pro testované parametry dosaženo. Pro srovnání jsou na obrázcích 5.13, 5.14, 5.15, 5.16 a 5.17 ukázány obrazy originální, poškozené a upravené pomocí editoru Lightroom. Hodnota PSNR byla vypočítána také pro editorem upravené obrazy a je uvedena v popisu obrázků.



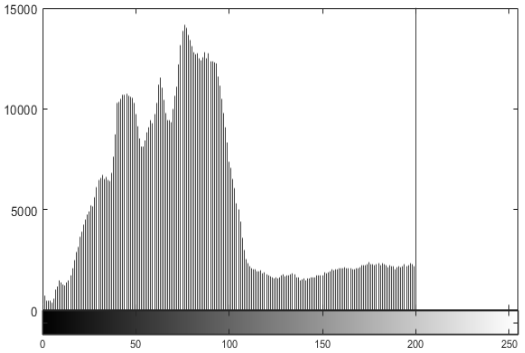
Obrázek 5.12: Ukázka změny parametrů při autokorekci obrazu 5.5 (a) editorem Adobe Photoshop Lightroom CC

(a) originál



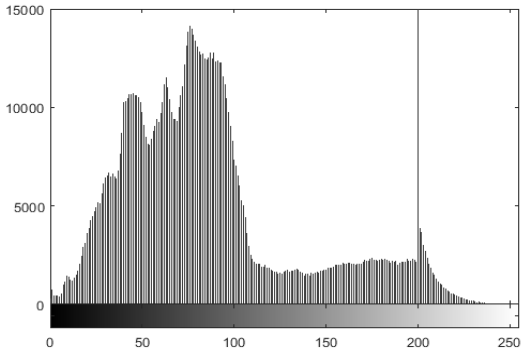
(b)

(c) poškozený obraz



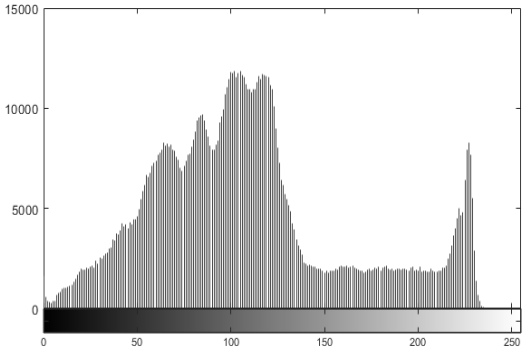
(d)

(e) restaurovaný



(f)

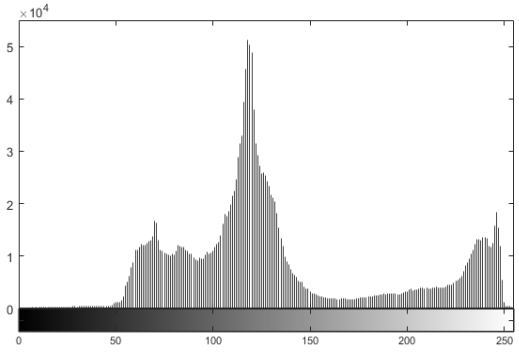
(g) editor Lightroom



(h)

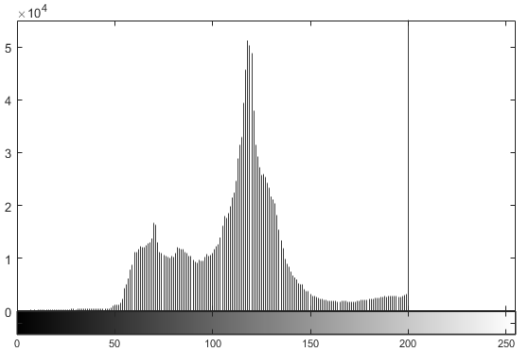
Obrázek 5.13: Originální obraz (5.5 (a)), poškozený obraz, obraz upravený navrženou metodou (vlnka = db10; úroveň dekompozice = 4; PSNR = 35,2072 dB; počet iterací = 700) a obraz upravený v editoru Lightroom (PSNR = 21,3637 dB). Na pravé straně jsou histogramy těchto obrazů.

(a) originál



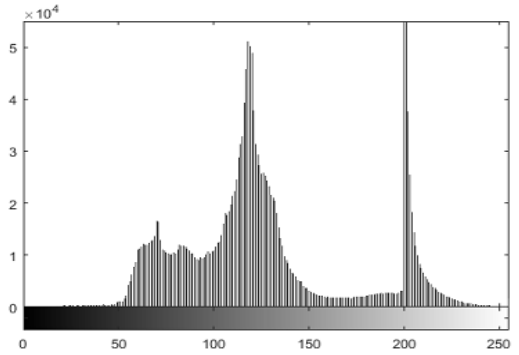
(b)

(c) poškozený obraz



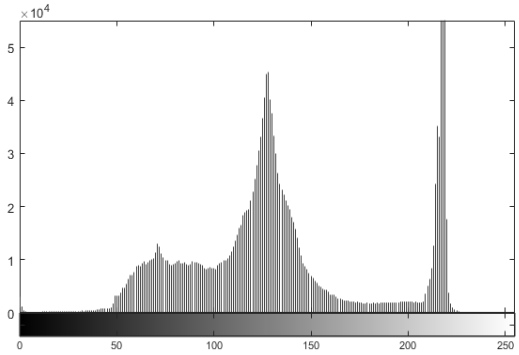
(d)

(e) restaurovaný



(f)

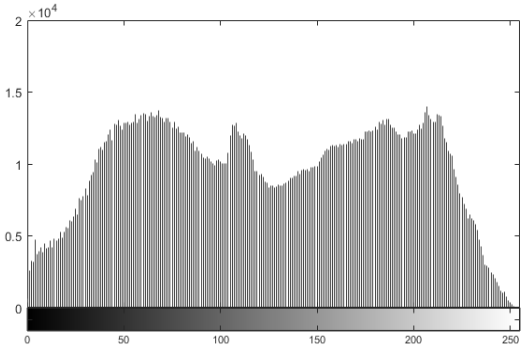
(g) editor Lightroom



(h)

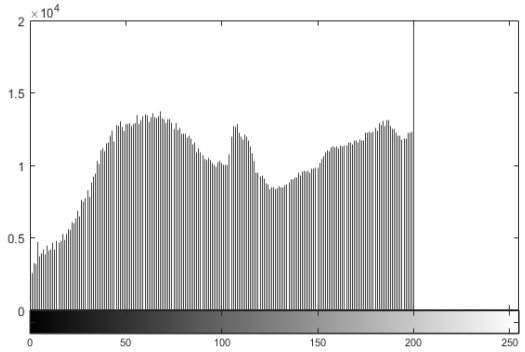
Obrázek 5.14: Originální obraz (5.5 (b)), poškozený obraz, obraz upravený navrženou metodou (vlnka = db40; úroveň dekompozice = 5; PSNR = 25,8416 dB; počet iterací = 700) a obraz upravený v editoru Lightroom (PSNR = 27,0122 dB). Na pravé straně jsou histogramy těchto obrazů.

(a) originál



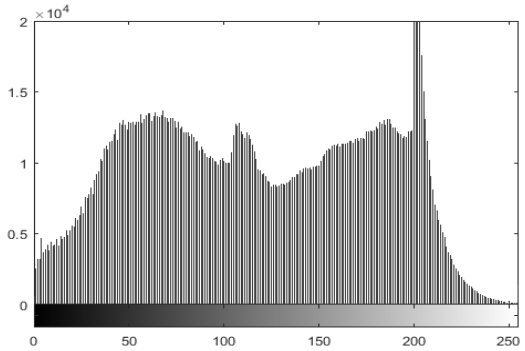
(b)

(c) poškozený obraz



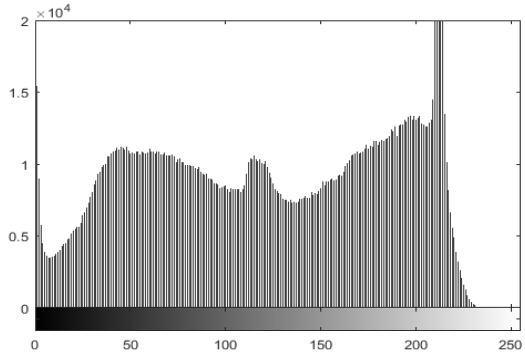
(d)

(e) restaurovaný



(f)

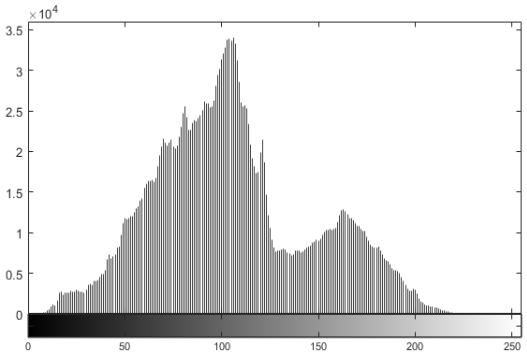
(g) editor Lightroom



(h)

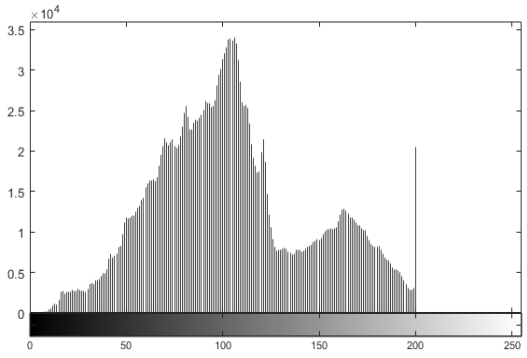
Obrázek 5.15: Originální obraz (5.5 (c)), poškozený obraz, obraz upravený navrženou metodou (vlnka = db30; úroveň dekompozice = 5; PSNR = 31,4638 dB; počet iterací = 439) a obraz upravený v editoru Lightroom (PSNR = 25,8477 dB). Na pravé straně jsou histogramy těchto obrazů.

(a) originál



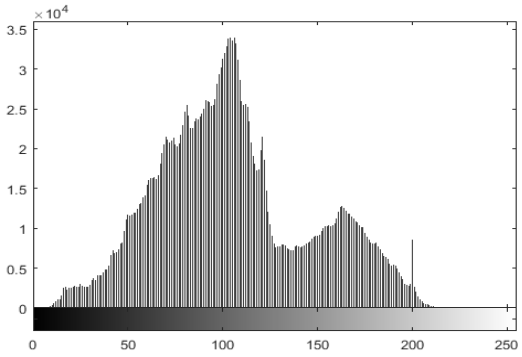
(b)

(c) poškozený obraz



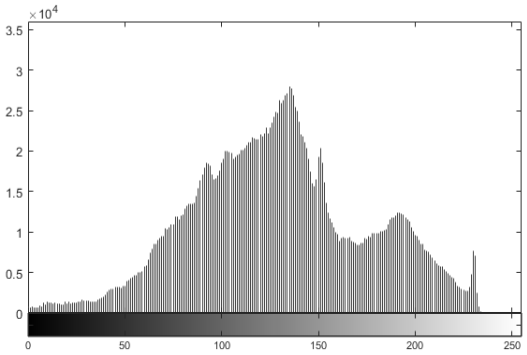
(d)

(e) restaurovaný



(f)

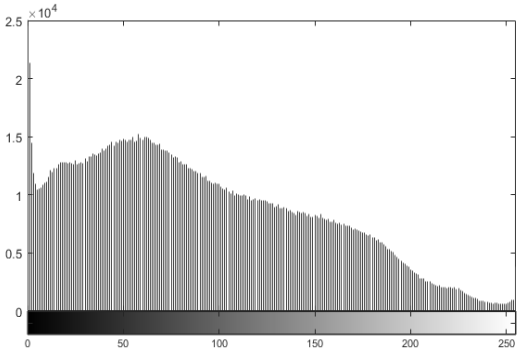
(g) editor Lightroom



(h)

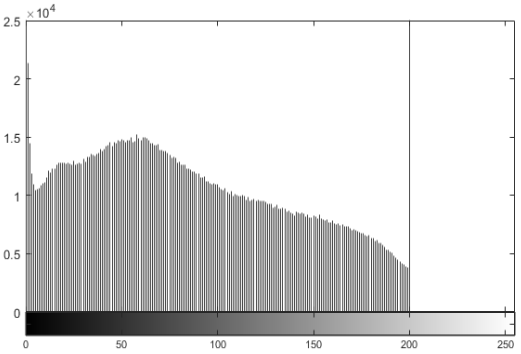
Obrázek 5.16: Originální obraz (5.5 (d)), poškozený obraz, obraz upravený navrženou metodou (vlnka = db40; úroveň dekompozice = 4; PSNR = 52,1017 dB; počet iterací = 227) a obraz upravený v editoru Lightroom (PSNR = 20,1128 dB). Na pravé straně jsou histogramy těchto obrazů.

(a) originál



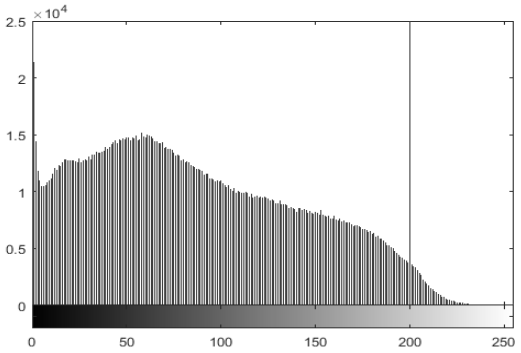
(b)

(c) poškozený obraz



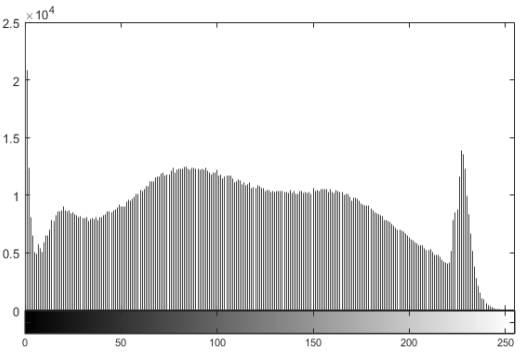
(d)

(e) restaurovaný



(f)

(g) editor Lightroom



(h)

Obrázek 5.17: Originální obraz (5.5 (e)), poškozený obraz, obraz upravený navrženou metodou (vlnka = db40; úroveň dekompozice = 5; PSNR = 34,8773 dB; počet iterací = 700) a obraz upravený v editoru Lightroom (PSNR = 19,5145 dB). Na pravé straně jsou histogramy těchto obrazů.

Porovnáním histogramů lze vidět, že navržená metoda pro všechny obrazy zachovala hodnoty pixelů, jež nebyly poškozeny a vypočítala hodnoty větší než θ (větší než maximální hodnota pixelu v obraze) v poškozených částech (histogram sice nepodává informace o tom, kde se dané pixely nacházejí, je však nepravděpodobné, že by se histogramy v části od 0 do θ shodovaly a při tom by hodnoty pixelů byly na jiném místě). Nicméně zde stále zůstává množství pixelů, jež si zachovaly hodnotu θ a nepřiblížily se k hodnotám originálního obrazu. Restaurované pixely se objevují nejvíce v okolí hodnoty θ a směrem k vyšším hodnotám jich ubývá. To je možné vidět i na obrázku 5.14, kde se obraz podařilo do jisté míry restaurovat, vytvořily se však shluky výrazně světlejších míst, která narušují vzhled obrazu. To je pravděpodobně způsobeno velkým množstvím pixelů, jež měly hodnoty nad hodnotu θ a povahou obrazu, kde se nacházejí velké světlé plochy. Pro obrázek 5.5 (d) bylo dosaženo poměrně vysoké hodnoty PSNR (v porovnání s ostatními obrazy) z toho důvodu, že hodnot, které byly poškozeny (jsou v originálním obraze větší než θ) není mnoho a tedy velká část obrazu se shoduje s obrazem referenčním. Navíc z obrázku 5.16 lze vidět, že histogram restaurovaného obrazu se blíží histogramu originálního obrazu (ale nepodává informace o tom, zda jsou tyto pixely na stejném místě jako u obrazu originálního).

Pro editorem upravené obrazy je možno z histogramu vyčíst, že nezachovává hodnoty pixelů nepoškozených míst a hodnoty pixelů v poškozené části se nerovnají histogramu originálního obrazu. Avšak vizuální dojem z obrazů může být lepší než v případě navržené metody (např. 5.14, kde nevznikají shluky světlých míst). Je také možné v editoru Adobe Photoshop (společnosti Adobe Systems) zvolit variantu, kdy označíme pouze poškozené pixely a funkcí Content-Aware Fill dopočítáme jejich novou hodnotu. Funkce zohledňuje především pixely nacházející se v okolí upravované oblasti a z nich doplňuje a upravuje označenou oblast. Tato varianta však nebyla zvolena z důvodu, že pro dosažení rozumných výsledků vyžaduje každý obraz subjektivní úpravu. Například u obrazu 5.5 (b) je velká plocha poškozených pixelů, vedle kterých se hned nachází pozadí. Pokud by nebyla zvolena oblast, ze které se doplňují poškozené pixely, jsou tyto pixely nahrazeny hodnotami podobnými pozadí, nikoliv daného objektu (což je možné vidět na obrázku 5.18). To by vedlo ke snížení objektivity a záleželo by spíše na intuici a zkušenosti editora, než na výpočtu editoru. Z důvodu časové náročnosti nebyla navržená metoda srovnána s jinými metodami, které vyžadují nastudování dané problematiky a následné naprogramování.

Obrázek 5.18: Ukázka úpravy obrazu 5.5 (b) v Adobe Photoshop pomocí funkce Content-Aware Fill bez úpravy oblasti, ze které se doplňují hodnoty pixelů

6 Závěr

Tato bakalářská práce byla zaměřena na problematiku restaurace přeexponovaných digitálních obrazů. Toho je snaha docílit díky předpokladu, že pro většinu obrazů lze najít řídké vyjádření tohoto obrazu. Je hledáno takové řídké vyjádření poškozeného obrazu při určitých požadavcích na hodnoty pixelů, které se blíží k obrazu referenčnímu.

V první části práce je popsán princip digitální fotografie a příčiny vzniku přeexponovaných (přepálených) fotografií včetně možností, jak se lze takovému poškození vyhnout. V další části je rozebrán teoretický úvod nutný pro návrh metody. Je zde definována norma vektoru a báze vektorového prostoru, které mají význam pro řídké vyjadřování signálu. Je uvedeno také řídké řešení systémů lineárních rovnic a jak takové řešení hledat pomocí konvexní optimalizace. Dále je zmíněna l_1 relaxace normy, která se s výhodou dá použít pro splnění podmínky konvexnosti. Indikátorová funkce je uvedena pro umožnění převodu omezené úlohy na úlohu neomezenou. Nechybí zde popis Douglas-Rachford algoritmu, pro který je definován proximální operátor a představena jeho podoba pro l_1 normu. V závěru této části je popsána vlnková transformace, jenž analyzuje signály jak ve frekvenční tak časové oblasti a dokáže poskytnout řídké vyjádření přirozených obrazů.

Následuje návrh metody, kde je formulována úloha jako konvexní optimalizační problém, který je řešen Douglas-Rachford algoritmem. Ten využívá dvou proximálních operátorů zohledňujících na jednu stranu požadavek na relaxovanou řídkost dat, na stranu druhou požaduje rovnost hodnot pixelů v nepoškozených částech obrazu s původním obrazem a v poškozených částech obrazu požaduje hodnoty pixelů větší nebo rovny maximální hodnotě pixelu poškozeného obrazu. Obsahem práce je také popis realizace takto navržené metody v programovém prostředí MATLAB.

V poslední části byla tato metoda testována pro různé parametry algoritmu (úroveň dekompozice 2-5, některé vlnky typu Daubechies, Symlets, Coiflets) a hodnocena na základě hodnoty PSNR (špičkový odstup signálu k šumu). Obrazy s nejlepší hodnotou PSNR jsou zde ukázány včetně jejich histogramů a to v porovnání s obrazy (i histogramy) originálními, poškozenými a upravenými v editoru fotografií Adobe Photoshop Lightroom CC, pro který byla také spočítána hodnota PSNR.

Metoda je navržena pro šedotónové obrazy (případně barevné, jež budou programem na šedotónové převedeny). Z grafů (obrázky 5.6, 5.7, 5.8, 5.9, 5.10) lze vidět, že pro různé obrazy je vhodné volit jiné parametry pro dosažení co nejlepšího výsledku. Pro všechny obrazy bylo dosaženo nejlepších výsledků pro úroveň dekompozice 4 nebo 5 a vlnky řádu 10 až 40. Na rozmanitější obrazy je zřejmě vhodné volit vyšší úroveň dekompozice i vlnky vyššího řádu.

Výhodou metody navržené v této práci je, že doplňuje hodnoty pixelů v místech, kde je obraz saturován a zároveň zachovává hodnoty obrazu v místech, která poškozená nejsou. Jiná situace je u autokorekce editoru typu Adobe Photoshop Lightroom CC (nebo například online editoru Polarr, který je snadno dostupný), jež tyto hodnoty mění, nicméně se snaží zlepšit vizuální dojem z fotografie.

Pro úpravu obrazů, u kterých nám nezáleží na zachování původních dat může být vhodné použít editor. Navržená metoda je naopak vhodná pro obrazy, u kterých je požadováno zachování nepoškozených dat a je schopná do určité míry obraz restaurovat. Toto také potvrzují hodnoty PSNR testovaných obrazů, které pro metodu navrženou v této práci dosahují hodnot v rozmezí 25 dB až 52 dB, zatímco v případě editoru to bylo jen 19 dB až 27 dB. Nízké hodnoty PSNR dosažené editorem jsou však dány především tím,

že pracuje s celým obrazem a ne pouze s poškozenými pixely. Mezi výhody editoru patří jeho intuitivnější použití (v případě autokorekce není ani třeba žádných znalostí úpravy fotek). Výsledky úprav navíc vidíme téměř okamžitě. Oproti tomu metoda navržená v této práci je časově náročnější a navíc není jednoznačné, jak vhodně zvolit parametry. Vyžaduje tedy určitou znalost této problematiky a zároveň je žádoucí vyzkoušet více variant, čímž roste časová náročnost celého procesu.

Lze se domnívat, že pro některé obrazy (5.5 (b), (c), (e)) by bylo možné dosáhnout lepších výsledků pro námi netestované parametry, zejména pro vyšší úroveň dekompozice. Pro zlepšení vizuálního dojmu obrazu se metoda nejeví jako příliš vhodná pro všechny obrazy. Například obraz (5.5 (b)) se sice do jisté míry restauroval, avšak vytvořily se zde shluky pixelů světlejší barvy, které vizuálně narušují vzhled obrazu. Záleží tedy na účelu rekonstrukce obrazu, zda-li jsou pro nás výsledky vhodné.

Literatura

[1] RAJMIC, Pavel a Marie DAŇKOVÁ. *Úvod do řídkých reprezentací signálů a komprimovaného snímání* [online]. Brno: Vysoké učení technické v Brně, 2014 [cit. 2019-04-22]. ISBN 978-80-214-5169-8. Dostupné z: http://www.utko.feec.vutbr.cz/ rajmic/skripta/Skripta-Ridke_reprezentace-Rajmic_Dankova.pdf

[2] PONEC, Jan a Milič JIRÁČEK. *Digitální fotografie*. Olomouc: Univerzita Palackého, 2002. ISBN 80-244-0533-4.

[3] PIHAN, Roman a Ondřej NEFF. *Mistrovství práce s DSLR: Vše, co jste chtěli vědět o digitální zrcadlovce a nikdo vám to neuměl vysvětlit*. 3. vyd. Praha: Institut digitální fotografie, 2008. ISBN 80-903210-8-9.

[4] HYAN, Jaroslav. *Digitální fotografie*. Praha: JTH-SOFT, 1998. ISBN 80-238-3176-3.

[5] RAJMIC, Pavel. *Základy počítačové sazby a grafiky*. Brno: Vysoké učení technické v Brně, 2012. ISBN 978-80-214-4451-5.

[6] Expozice. In: *Online fotoškola Martina Krejčího* [online]. Praha: Martin Krejčí, c2013-2018 [cit. 2018-12-04]. Dostupné z: https://www.onlinefotoskola.cz/pomucky/databaze-fotografickych-pojmu/expozice.html

[7] Přeexpozice. In: *Online fotoškola Martina Krejčího* [online]. Praha: Martin Krejčí, c2013-2018 [cit. 2018-12-04]. Dostupné z: https://www.onlinefotoskola.cz/pomucky/databaze-fotografickych-pojmu/p%C5%99eexpozice.html

[8] Podexpozice. In: *Online fotoškola Martina Krejčího* [online]. Praha: Martin Krejčí, c2013-2018 [cit. 2018-12-04]. Dostupné z: https://www.onlinefotoskola.cz/pomucky/databaze-fotografickych-pojmu/podexpozice.html

[9] Přepal. In: *Online fotoškola Martina Krejčího* [online]. Praha: Martin Krejčí, c2013-2018 [cit. 2018-12-04]. Dostupné z: https://www.onlinefotoskola.cz/pomucky/databaze-fotografickych-pojmu/přepal.html

[10] Přepal, přepálená bílá. In: *FotoRomán* [online]. Roman Pihan, c2002-2017 [cit. 2018-12-04]. Dostupné z: http://www.fotoroman.cz/glossary/3_prepal.htm

[11] Přepálené fotografie? Pochopte, co je dynamický rozsah, a jsou minulostí. In: *Milujeme fotografii od Zoner Photo Studia* [online]. Brno: ZONER software, a.s, 2016 [cit. 2018-12-04]. Dostupné z: https://www.milujemefotografii.cz/prepalene-fotografie-pochopte-co-je-dynamicky-rozsah-a-jsou-minulosti

[12] Jak se vypořádat s přepaly a podpaly. In: *Fotorádce* [online]. Praha: Jan Šmíd, 2013 [cit. 2018-12-04]. Dostupné z: https://www.fotoradce.cz/jak-se-vyporadat-s-prepaly-a-podpaly

[13] Technika HRD v praxi. In: *Milujeme fotografii od Zoner Photo Studia* [online]. Brno: ZONER software, a.s, 2013 [cit. 2018-12-04]. Dostupné z: https://www.milujemefotografii.cz/technika-hdr-v-praxi

[14] HDR fotografie. In: *Canon* [online]. Praha: Canon CZ, c2018 [cit. 2018-12-04]. Dostupné z: https://www.canon.cz/get-inspired/come-and-see/showcase/hdr-photography/

[15] HUTCHINSON, John E., LOY, R. J., ed. *Introduction To Mathematical Analysis* [online]. Mathematics School of Mathematical Sciences ANU, 1994 [cit. 2019-04-21]. Dostupné z: https://maths-people.anu.edu.au/ john/Assets/Lecture%20Notes/B21H_97.pdf

[16] HRBÁČEK, Radek, Pavel RAJMIC, Vítězslav VESELÝ a Jan ŠPIŘÍK. Řídké reprezentace signálů: úvod do problematiky. *Elektrorevue* [online]. 2011, 13. 9. 2011, (5) [cit. 2019-04-21]. ISSN 1213-1539. Dostupné z: http://www.elektrorevue.cz/cz/clanky/zpracovani-signalu/0/ridke-reprezentace-signalu–uvod-do-problematiky/

[17] CANDÈS, Emmanuel J., Michael B. WAKIN a Stephen P. BOYD. Enhancing Sparsity by Reweighted - 1 Minimization. *Journal of Fourier Analysis and Applications* [online]. 2008, **14**(5-6), 877-905 [cit. 2019-04-12]. DOI: 10.1007/s00041-008-9045-x. ISSN 1069-5869. Dostupné z: https://web.stanford.edu/ boyd/papers/pdf/rwl1.pdf

[18] ŠPIŘÍK, Jan, Pavel RAJMIC a Vítězslav VESELÝ. Reprezentace signálů: od bází k framům. *Elektrorevue* [online]. 2010, (6) [cit. 2019-04-22]. ISSN 1213-1539. Dostupné z: http://www.elektrorevue.cz/cz/clanky/zpracovani-signalu/35/reprezentace-signalu–od-bazi-k-framum/

[19] DOŠLÁ, Zuzana a Jaromír KUBEN. *Diferenciální počet funkcí jedné proměnné*. 2. vyd. Brno: Masarykova univerzita, 2012. ISBN 978-80-210-5814-9.

[20] COMBETTES, Patrick L. a Jean-Christophe PESQUET. *Proximal Splitting Methods in Signal Processing* [online]. , 1-10 [cit. 2019-04-01]. Dostupné z: https://arxiv.org/pdf/0912.3522.pdf

[21] DAŇKOVÁ, Marie. *Komprimované snímání v perfuzním zobrazování pomocí magnetické rezonance*. Brno, 2014. Diplomová práce. Vysoké učení technické v Brně.

[22] BACHMAYR, Markus a Reinhold SCHNEIDER. Iterative Methods Based on Soft Thresholding of Hierarchical Tensors. *Foundations of Computational Mathematics* [online]. New York: Springer US, 2017, **17**(4), 1037-1083 [cit. 2019-04-11]. DOI: 10.1007/s10208-016-9314-z. ISSN 1615-3375. Dostupné z: https://link-springer-com.ezproxy.lib.vutbr.cz/article/10.1007/s10208-016-9314-z

[23] ŠVEC, Martin. *Waveletové transformace*. Ústí nad Labem: Univerzita J. E. Purkyně v Ústí nad Labem, Přírodovědecká fakulta, 2008. ISBN 978-80-7044-987-5.

[24] MALLAT, Stéphane. *A Wavelet Tour of Signal Processing: The Sparse Way*. With contributions from Gabriel Peyré. 3rd ed. United States: Elsevier, c2009. ISBN 978-0-12-374370-1.

[25] ANISIMOVA, Elena, Jan BEDNÁŘ a Petr PÁTA. Zpracování obrazu pomocí vlnkové transformace. *Elektrorevue* [online]. 2013, (4), 237-246 [cit. 2019-04-15].

ISSN 1213-1539. Dostupné z: http://www.elektrorevue.cz/cz/clanky/zpracovani-signalu/10/zpracovani-obrazu-pomoci-vlnkove-transformace–image-processing-using--the-wavelet-transform-/

[26] HORA, Petr. *Waveletová analýza: Díl I. - Teoretický úvod*. Plzeň: Západočeská univerzita v Plzni, 1996.

[27] LEE, Daniel T. L. a Akio YAMAMOTO. Wavelet Analysis: Theory and Applications. *Hewlett-Packard Journal* [online]. 1994, , 44-52 [cit. 2019-05-09]. Dostupné z: https://www.hpl.hp.com/hpjournal/94dec/dec94a6a.pdf

[28] HORÁK, David. *Diskrétní transformace* [online]. VŠB — Technická univerzita Ostrava a Západočeská univerzita v Plzni, 2012 [cit. 2019-05-12]. Dostupné z: http://mi21.vsb.cz/sites/mi21.vsb.cz/files/unit/diskretni_transformace.pdf

[29] MALÝ, Jan. Srovnání metod pro ztrátovou kompresi obrazu. *Elektrorevue* [online]. 2006 [cit. 2019-05-05]. Dostupné z: http://www.elektrorevue.cz/cz/clanky/zpracovani-signalu/0/srovnani-metod-pro-ztratovou-kompresi-obrazu/

[30] SUSU YAO, WEISI LIN, EEPING ONG a ZHONGKANG LU. Contrast signal-to-noise ratio for image quality assessment. In: *IEEE International Conference on Image Processing 2005* [online]. IEEE, 2005, 2005, I-397 [cit. 2019-05-05]. DOI: 10.1109/ICIP.2005.1529771. ISBN 0-7803-9134-9. Dostupné z: http://ieeexplore.ieee.org/document/1529771/

[31] Wavelet Families. In: *The MathWorks* [online]. Massachusetts: Little, c1994-2019 [cit. 2019-05-08]. Dostupné z: https://www.mathworks.com/help/wavelet/ug/wavelet-families-additional-discussion.html#f8-45577

A Dodatek

Algoritmus se spouští ze skriptu `startAlg`, kde se do `vtup` zapíše název žádaného obrázku včetně přípony, který se nachází ve stejné složce. Dále se nastaví parametry Douglas-Rachford algoritmu, tedy lambda `lambda` při výpočtu $\mathbf{y}_{n+1}$, `gama` jakožto práh pro měkké prahování, typ waveletu `wavelet`, úroveň dekompozice `uroven`, maximální počet iterací `MaxIt` a tolerance pro ukončení algoritmu `delta`.

Je možné nastavení hodnot `prah` a `prahH` pro přípravu chybného obrazu. Po nastavení všech hodnot algoritmus volá funkci `pripPic`, která načte a převede na šedotónový originální obraz `Pic`, vytvoří chybný obraz `PicKlip` a obrazy `bH` a `bL` pro výpočet proximálního operátoru. Následně ve funkci `Doug_Rach` probíhá cyklus výpočtu pro restauraci obrazu, kde se počítají dva proximální operátory a nový obraz. Po dokončení tohoto cyklu se nově spočítaný obraz zpracovává ve funkci `vysledek`, kde se vykreslí originální obraz `Pic`, restaurovaný obraz `vysledny0br` a poškozený obraz `PicKlip` včetně jejich příslušných histogramů.