# Fraktální komprese statických obrazů pomocí waveletové transformace

Fractal image compression using wavelet transformation

Bc. Jiří Giesl

Diplomová práce
2007

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav aplikované informatiky
akademický rok: 2006/2007

# ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

| | |
|---|---|
| Jméno a příjmení: | **Bc. Jiří GIESL** |
| Studijní program: | **N 3902 Inženýrská informatika** |
| Studijní obor: | **Informační technologie** |
| Téma práce: | **Fraktální komprese statických obrazů pomocí waveletové transformace** |

Zásady pro vypracování:

1. **Vypracování literární rešerše na dané téma.**
2. **Analýza vlastností waveletové transformace včetně nalezení bázových funkcí vhodných pro použití při fraktální kompresi (dále FIC).**
3. **Návrh řešení systému pro FIC a jeho implementace.**
4. **Optimalizace řešení systému FIC.**
5. **Experimentální vyhodnocení a srovnání s jinými kompresními metodami.**

Rozsah práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

**Levický, D.: Multimediálne komunikácie, ELFA, Košice (2002), ISBN 80-89066-58-5**
**Zelinka Ivan: Differential Evolution Global Optimization for Scientists and Engineers, Springer-Verlag, chap. 5.5.1 "Inverse Fractal Problem", 13 p. (in print)**

Vedoucí diplomové práce: **prof. Ing. Karel Vlček, CSc.**
Ústav aplikované informatiky

Datum zadání diplomové práce: **13. února 2007**

Termín odevzdání diplomové práce: **28. května 2007**

Ve Zlíně dne 13. února 2007


prof. Ing. Vladimír Vašek, CSc.
*děkan*

doc. Ing. Ivan Zelinka, Ph.D.
*ředitel ústavu*

## ABSTRAKT

Fraktální komprese nepatří mezi standardizované metody. Existuje spousta metod a algoritmů, které různě využívají významnou vlastnost fraktálů, sobě-podobnost. Tato práce popisuje waveletovou transformaci jako pomocný nástroj pro zpracování signálů a kompresi dat. Samotná komprese, která je zde navržena, je založena na principech fraktální komprese obrazu po provedení redukce nadbytečných informací pomocí diskrétní waveletové transformace a hledáním sobě-podobnosti waveletových koeficientů v nízko-frekvenčním pásmu dyadicky rozloženého 2D signálu.

Klíčová slova: fraktální komprese, sobě-podobnost, waveletová transformace, zpracování signálů, komprese dat

## ABSTRACT

Fractal compression belongs to non-standardized methods. There are more methods and algorithms, which are using the important feature of fractals, self-similarity. This work deals with wavelet transformation as a support tool for signal processing and data compression. Compression here presented is based on principles of fractal compression of image after the reduction of redundant information by wavelet transformation and by searching for self-similarity of wavelet coefficients in low-frequency sub-band of dyadic decomposed 2D signal.

Keywords: fractal compression, self-similarity, wavelet transformation, image processing, data compression

Rád bych na tomto místě poděkoval vedoucímu mé diplomové práce panu prof. Ing. Karlu Vlčkovi, CSc. za jeho vstřícnost a ochotu. Děkuji také všem lidem, kteří mi byli nápomocni po celé studium, a hlavně své rodině, která mi byla oporou celý můj dosavadní život.

Murphyho zákon: *„Nejvíce programových chyb vzniká mezi klávesnicí a židlí.“*

Prohlašuji, že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků, je-li to uvolněno na základě licenční smlouvy, budu uveden jako spoluautor.

Ve Zlíně dne 21.5.2007

…….
Podpis diplomanta

# OBSAH

# ÚVOD

Fraktální komprese je postup, jak nalézt a odstranit nadbytečné informace ve formě podobných, opakujících se vzorů ve statickém obrazu či jiném multimediálním signálu. Prací na toto téma bylo již napsáno mnoho. Většina z nich se zabývala optimalizací rychlosti kódovacích algoritmů, protože tento typ komprese je na rozdíl od ostatních silně asymetrický, nebo také zajištěním kvalitnějšího výstupu na stejném či podobném kompresním poměru. Vzniklo tak tedy spousta různých variant fraktální komprese s různými vlastnostmi a kvůli tomu není dodnes vytvořen standardizovaný postup a pravděpodobně vytvořen ani nebude. Je možné, že právě z tohoto důvodu nebude fraktální komprese nikdy využívána veřejností ke každodenním účelům.

V oblasti digitálního zpracování dat se ukázalo, že pro analýzu nestacionárních signálů, mezi něž patří většina multimediálních signálů, je zatím nejvhodnějším nástrojem waveletová transformace. Pomocí ní lze převést signál do časově-frekvenční oblasti, ve které se další analýzy provádí daleko efektivněji. Signál lze v této oblasti jednodušeji klasifikovat a rozlišovat, které složky signálu jsou více či méně redundantní. Díky tomu je možné provádět se signály mnoho operací, jako např. zbavovat šumu či dokonce také komprimovat.

Spojením fraktální komprese s vlastnostmi waveletové transformace je tedy teoreticky možné implementovat kodér, se kterým se může docílit velmi dobrých výsledků. Tato diplomová práce se zabývá právě tímto kodérem, kdy po waveletové analýze statického obrazu se využívá fraktální komprese pro hledání sobě-podobnosti frekvenčních složek. Na základě popisu této sobě-podobnosti je celý obraz komprimován. Pomocí waveletové transformace se správně zvolenou bázovou funkcí se poté docílí i dobrých výsledků při samotné rekonstrukci statického obrazu.

V následujících kapitolách je celý postup podrobněji vysvětlen. Kromě základní teorie číslicových filtrů je popsána i samotná waveletová transformace, u níž je především zaměřeno na její diskrétní formu. Následuje rozbor fraktální geometrie a její využití pro kompresní účely. Experimentální výsledky tohoto fraktálního kodéru jsou objektivně hodnoceny PSNR ukazatelem a srovnávány s výstupy ztrátové kompresní metody JPEG.

# I. TEORETICKÁ ČÁST

# 1 LITERÁRNÍ REŠERŠE

V současné době není mnoho odborných publikací, které by se zabývaly fraktální kompresí komplexně. Přesto lze na Internetu a webových stránkách určených pro odbornou veřejnost nalézt nepřeberné množství článků a vědeckých příspěvků, které prezentují výsledky různých variant fraktální komprese a diskutují teoretický problém.

Pravděpodobně nejznámější publikací věnované fraktální geometrii je *The fractal geometry of nature* od Benoita Mandelbrota. Autor v ní definuje, co je to fraktál, jaké jsou jeho vlastnosti, a že je lze vytvořit pomocí velmi malých konečných algoritmů a dat.

Knižní publikace *Fractals Everywhere* od Michaela Barnsleye se zabývá tématem, jak fraktální geometrii použít k popisu objektů v reálném světě. Objevuje se zde tedy významná definice IFS (systémů iterovaných funkcí) a je uvedeno, jaké musí mít tyto systémy vlastnosti, aby pomocí nich bylo možné reprezentovat např. statický obraz.

Významnou monografií zabývající se fraktální kompresí na použitelné úrovni je *Fractal Image Compression: Theory and Application* od Yuvala Fischera. Kniha obsahuje několik článků o fraktální kompresi od odborníků, kteří se touto problematikou zabývají. Je zde prezentována teorie a implementace metod fraktální komprese založené na hledání sobě-podobných částí obrazu. Kromě jednoduchého fraktálního kodéru je zde především prezentována metoda Quadtree Partitioning, kdy jsou části obrazu rekurzivně děleny do určité úrovně. Dále se také zabývá urychlujícími prostředky kódovacího procesu a snahou dosáhnout vysokých kompresních poměrů. Následuje srovnávání různých fraktálních schémat s používanými kompresními metodami. Metody fraktální komprese, které jsou zde zmíněny, jsou z hlediska matematiky popsány pouze nezbytnými matematickými vztahy a autor zde předkládá především implementační záležitosti.

Prací, která může být přínosná především pro začátečníky v oboru fraktální komprese, je *A Hitchhiker's Guide to „Fractal-Based" Function Approximation and Image Compression* od Edwarda Vrscaye. Jedná se o příručku, která popisuje, jak jsou IFS využívány pro reprezentaci a kompresi obrazu. Zabývá se tou nejzákladnější formou fraktální komprese, od níž jsou ostatní varianty, které jsou dnes vyvíjeny, odvozeny.

K rozšíření znalostí o fraktální kompresi je výborná publikace *Fractal Image Compression: An Introductory Overview* od D. Saupeho, R. Hamzaouiho a H. Hartensteina. Je zde popsána vektorová kvantizace a základní fraktální kodér. Dále jsou

prezentovány základní myšlenky adaptivního Quadtree kodéru Yuvala Fischera. Zaměřeno je také především na snížení výpočetní náročnosti kódování a jsou tedy popsány urychlující operace a některá klasifikační schémata. Nastíněny jsou i pokročilé techniky hierarchického dělení obrazu jako je polygonální, horizontálně-vertikální dělení nebo dělení pomocí evolučního algoritmu. V posledních kapitolách jsou zmíněny některé hybridní metody fraktálního kódování.

Příspěvek Alexe van de Walleho nazvaný *Merging Fractal Image Compression And Wavelet Transform Methods* popisuje fraktální kompresi ve frekvenční oblasti, která se získala použitím waveletové transformace. Kromě teoreticky nastíněného algoritmu jsou ukázány výhody waveletové reprezentace. Zajímavou podkapitolou je ukázání způsobu, jak urychlit hledání a srovnávání příslušných waveletových koeficientů při hledání sobě-podobnosti.

Zajímavým článkem, který popisuje fraktální kodér využívající waveletovou transformaci je *A Wavelet-Based Analysis of Fractal Image Compression* od Geoffrey Davise. Kromě klasické fraktální komprese popisuje i fraktální kompresi ve waveletové oblasti a využití tzv. wavelet subtrees, což jsou sady koeficientů reprezentující hierarchickou datovou strukturu v dyadicky rozloženém obrazu. Tyto subtrees se používají poté pro hledání sobě-podobných částí obrazu. Dále je popsáno i kódovací SQS (self-quantization of subtrees) kvantizační schéma, pomocí kterého lze dosáhnout velmi dobrých výsledků i na vysokých kompresních poměrech.

Hybridní fraktální kodér popsaný v článku *A List Directed Approach to Fractal Image Coding in the Wavelet Transform Domain*, který zveřejnili I. Messing a D. Malah, využívá Quadtree Partitioning pro získání různě velkých wavelet subtrees, které seřazují do seznamu a u kterých se následně hledá sobě-podobnost.

Příspěvek prezentovaný čtveřicí vědců L.M. Poem, Y. Zhangem, K.W. Cheungem a C.H. Cheungem je nazván *A Novel Subtree Partitioning Algorithm for Wavelet-Based Fractal Image Coding* a představuje fraktální kodér, který rozděluje subtree na skalárně kvantizované waveletové koeficienty a fraktálně kódované sub-subtree. Tato metoda dělení a kódování poskytuje ještě o něco lepší výsledky než SQS schéma od Geoffreyho Davise a navíc je méně početně náročná.

# 2 TEORIE DIGITÁLNÍCH FILTRŮ

## 2.1 Digitální filtry

Filtr je chápán jako časově nezávislý operátor, který ovlivňuje frekvenční charakteristiku systému a lze zapsat jako:

$$y(n) = \sum_{k} h(k) \cdot x(n-k) \tag{2.1}$$

kde $x$ je vektor vstupního signálu, $y$ je vektor výstupního signálu a $h$ je vektor koeficientů, které charakterizují přenosovou charakteristiku filtru.

### 2.1.1 Filtr typu dolní propust

Filtr typu dolní propust se používá k analýze nižších frekvencí signálu a může být popsán vztahem:

$$y_{LP}(n) = \frac{1}{2} x(n) + \frac{1}{2} x(n-1) \tag{2.2}$$

Tento typ filtru se nazývá také klouzavý průměr a vyhlazuje rychlé změny v signálu.

### 2.1.2 Filtr typu horní propust

Filtr typu horní propust se používá k analýze vyšších frekvencí signálu a lze popsat vztahem:

$$y_{HP}(n) = \frac{1}{2} x(n) - \frac{1}{2} x(n-1) \tag{2.3}$$

Tento typ filtru se nazývá také klouzavý rozdíl a zachycuje rychlé změny v signálu.

## 2.2 Podvzorkování a převzorkování

Při návrhu filtrů a jejich aplikací hraje důležitou roli vzorkování. Při podvzorkování (downsampling) není dodržen Shannon-Kotělnikovův teorém a znamená to, že se sníží počet prvků vstupní posloupnosti $x(n)$ na polovinu. Při převzorkování (upsampling) naopak dochází k doplnění chybějících vzorků, takže se počet prvků zdvojnásobí.

### 2.2.1 Podvzorkování (downsampling)

Signál, který byl podvzorkován, lze zapsat následujícím maticovým tvarem:

$$(\downarrow 2)\begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \\ x(4) \\ \dots \end{bmatrix} = \begin{bmatrix} x(0) \\ x(2) \\ x(4) \\ \dots \\ \dots \\ \dots \end{bmatrix} \tag{2.4}$$

kde operátor $(\downarrow 2)$ nazýváme lineárním operátorem podvzorkování číslem 2. Aplikací operátoru podvzorkování na vektor vstupního signálu získáme modifikovaný signál, ze kterého jsou všechny liché vzorky odstraněny.

Operátor podvzorkování může být maticově vyjádřen jako:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix} \tag{2.5}$$

Jak lze vidět, jednotlivé řádky matice podvzorkování jsou vzájemně ortonormální. Každý jednotlivý řádek představuje jednotkový vektor a jejich vzájemný skalární součin je roven nule.

### 2.2.2 Převzorkování (upsampling)

Signál, který byl převzorkován, může být zapsán následujícím maticovým tvarem:

$$(\uparrow 2)\begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ \dots \\ \dots \\ \dots \\ \dots \end{bmatrix} = \begin{bmatrix} x(0) \\ 0 \\ x(1) \\ 0 \\ x(2) \\ 0 \\ \dots \end{bmatrix} \tag{2.6}$$

kde operátor $(\uparrow 2)$ nazýváme lineárním operátorem převzorkování číslem 2. Aplikací operátoru převzorkování na vektor vstupního signálu získáme modifikovaný signál, v němž jsou všechny liché vzorky doplněny nulami.

Operátor převzorkování lze zapsat maticově jako:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots \end{bmatrix} \tag{2.7}$$

Na matici operátoru převzorkování lze vidět, že je tato matice transponovaná vzhledem k matici opeátoru podvzorkování, což dokazuje vztah:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \dots & \dots & \dots \end{bmatrix} \tag{2.8}$$

tedy:

$$(\downarrow 2)\cdot(\uparrow 2) = I \tag{2.9}$$

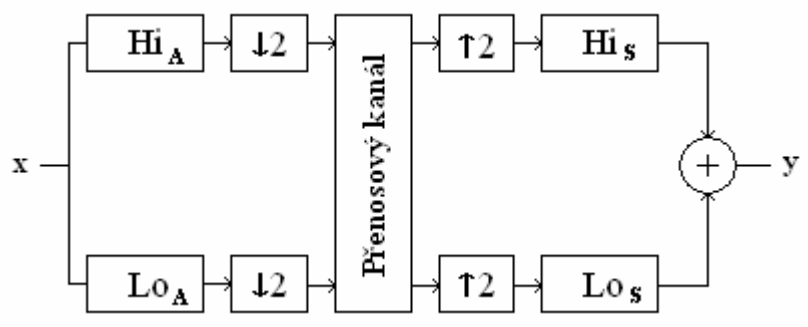
Vzhledem k platnosti vztahu (2.9) a na základě vlastnosti ortogonality tedy můžeme říci, že transponovaná matice podvzorkování je ekvivalentní matici převzorkování:

$$(\downarrow 2)^T = (\uparrow 2) \tag{2.10}$$

## 2.3 Banka filtrů

Banku filtrů lze definovat jako sadu filtrů typu horní propust a dolní propust. Oba dva typy mají vzájemně oddělené frekvenční přenosové charakteristiky. Součet těchto charakteristik tvoří celkovou přenosovou charakteristiku sytému.

Na Obr.1. je zobrazeno schéma zpracování signálu $x$ bankou filtrů.




Obr. 1. Analýza a syntéza signálu pomocí banky filtrů

Vstupní signál $x$ se nechá projít filtrem typu horní propust (blok $Hi_A$) k analýze vyšších frekvencí a filtrem typu dolní propust (blok $Lo_A$) k analýze nižších frekvencí. Po provedení podvzorkování (blok $\downarrow 2$) takto analyzovaného signálu je signál přenesen. Po přenosu je signál převzorkován (blok $\uparrow 2$) a je provedena jeho syntéza pomocí příslušných hornopropustních a dolnopropustních filtrů (bloky $Hi_S$ a $Lo_S$).

Úkolem při návrhu banky filtrů je dokonalá rekonstrukce signálu, tedy po provedení všech operací by mělo platit, že: $y = x$.

# 3 WAVELETOVÁ TRANSFORMACE

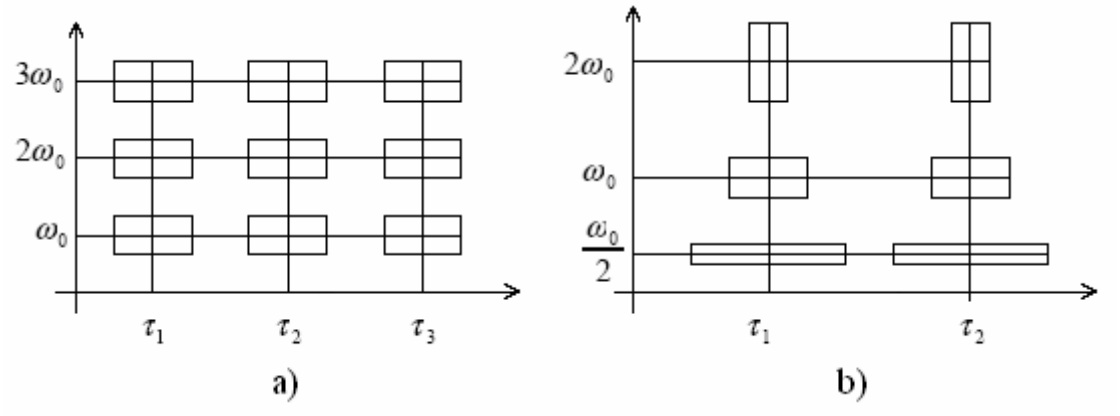
Při analýze a reprezentaci signálů je často výhodné použít transformaci, která reprezentuje signál i v jiné než časové (polohové) oblasti. Vhodně zvolenou transformací můžeme získat příznaky, které zjednoduší případnou klasifikaci signálu. Signál lze také před přenosem digitální sítí transformovat do mnohem výhodnější podoby z hlediska redundantní informace nebo citlivosti k přenosovým chybám a po ukončení přenosu tento signál transformovat zpět do původního tvaru.

## 3.1 Princip waveletové transformace

Za jednu z významných transformačních technik se dá považovat *Fourierova transformace*, která poskytuje informaci o tom, které frekvence se v signálu nacházejí. Nevypovídá však o jejich poloze v čase a proto je vhodná pouze ke frekvenčnímu popisu stacionárních signálů. Tento problém lze řešit zavedením okna, které v čase ohraničí krátký úsek signálu a pomocí něj lze určovat frekvenční spektrum v daném časovém intervalu. Tento postup se nazývá *Short-Time Fourier Transformation* (STFT). Z obdoby Heisenbergova principu neurčitosti vyplývá, že nelze současně určit přesně frekvenci a polohu jejího výskytu v čase. Proto má uvedené řešení pro časově konstantně široké okno pro všechny kmitočty velkou rozlišitelnost ve frekvenci a malou v čase a naopak pro časově úzké okno velkou rozlišitelnost v čase a malou ve frekvenci.

Ideou *waveletové transformace* (WT) je vhodnou změnou šířky okna v čase a jeho tvarem dosáhnout optimálního poměru rozlišitelnosti v čase a frekvenci. Pro nízké frekvence je okno širší, pro vysoké užší. Toto okno se nazývá mateřský wavelet $\psi$ (*mother wavelet*). [1]

Rozdíl mezi STFT a WT na Obr.2. je tedy na první pohled patrný.




Obr. 2. Časově-frekvenční oblast s rozložením oken pro a) STFT a b) WT

## 3.2 CWT

Definiční vztah pro spojitou waveletovou transformaci (Continuos WT) lze zapsat ve tvaru

$$W(s,\tau) = \int f(t)\psi^*_{s,\tau}(t)dt \tag{3.1}$$

kde $*$ je označením komplexně sdružené proměnné. Tento vztah reprezentuje rozložení funkce $f(t)$ do sady bázových funkcí $\psi_{s,\tau}(t)$, které budeme nazývat *wavelety*.

Wavelety jsou generovány z jediné bázové funkce $\psi(t)$, tzv. *mateřského waveletu*

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{|s|}}\psi\left(\frac{t-\tau}{s}\right) \tag{3.2}$$

Pomocí parametru $s$, který se jmenuje měřítko, je možné měnit šířku waveletu (dilatace). Člen $\sqrt{|s|}$ je zde nutný z důvodu normalizace energie při změně měřítka. Tím je zajištěno, že wavelet bude mít pro všechna měřítka normalizovanou energii.

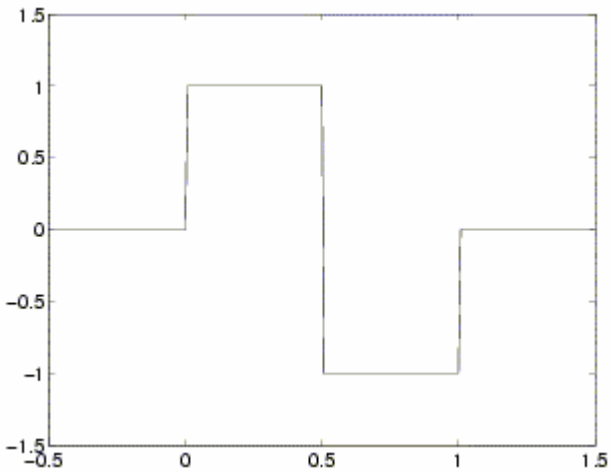
Parametrem $\tau$ zvaným poloha se mění umístění waveletu na časové ose (translace).

### 3.2.1 Typy mateřských waveletů

Volba vhodné bázové funkce ovlivňuje celkové výsledky transformace a tím také kvalitu analyzovaného signálu. Pro různé signály a aplikace byly navrženy různé mateřské wavelety.

#### *3.2.1.1 Haar wavelet*

Tato bázová funkce je jednoduše implementovatelná. Neumožňuje hladkou rekonstrukci signálu, ovšem na druhou stranu je symetrická a ortogonální a proto je vhodná jak pro spojitou (CWT), tak i diskrétní (DWT) waveletovou transformaci.

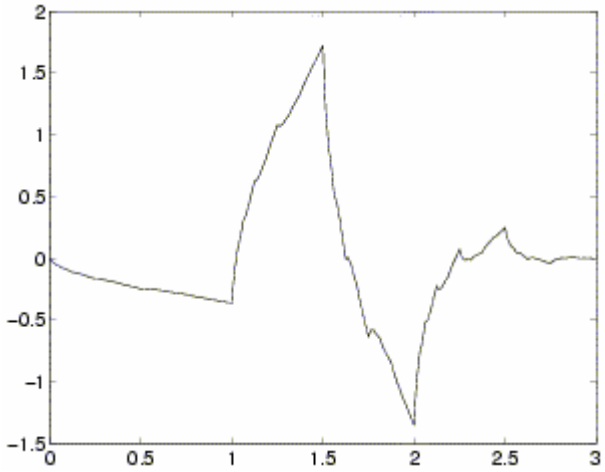



Obr. 3. Haar wavelet

$$\psi(x) = \begin{cases} 1 & 0 \le x < 1/2 \\ -1 & \text{pro} \quad 1/2 < x < 1 \\ 0 & jinde \end{cases} \tag{3.3}$$

#### *3.2.1.2 Daubechies wavelet*

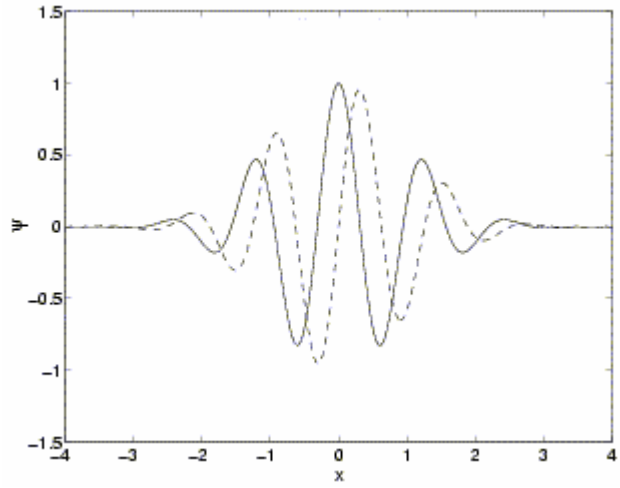
Vytváří skupinu bázových funkcí různých řádů. Nemají explicitní vzorec (kromě 1.řádu, který je ekvivalentní Haar waveletu). Jsou asymetrické a ortogonální a tedy stále vhodné pro CWT i DWT.




Obr. 4. Daubechies wavelet

### *3.2.1.3 Morlet wavelet*

Jedná se o mateřský wavelet s tvarem komplexní sinusovky modulované Gaussovým oknem. Je symetrická, komplexní, ovšem není ortogonální, tudíž se hodí pouze pro CWT.

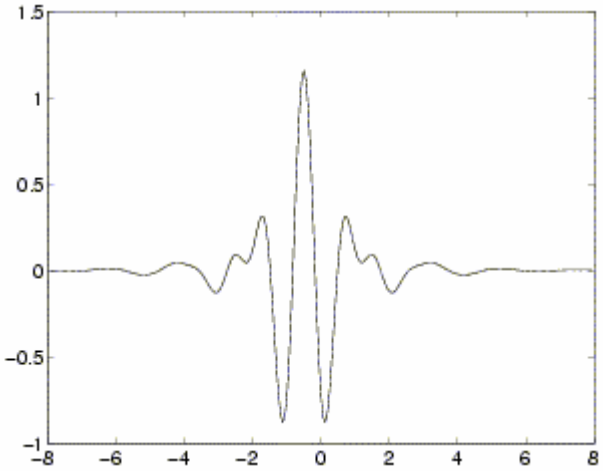



Obr. 5. Morlet wavelet

$$\psi(x) = a \cdot e^{-\lceil 1/2 \rfloor x^2} ( \cos(5x) + j \sin(5x)) \tag{3.4}$$

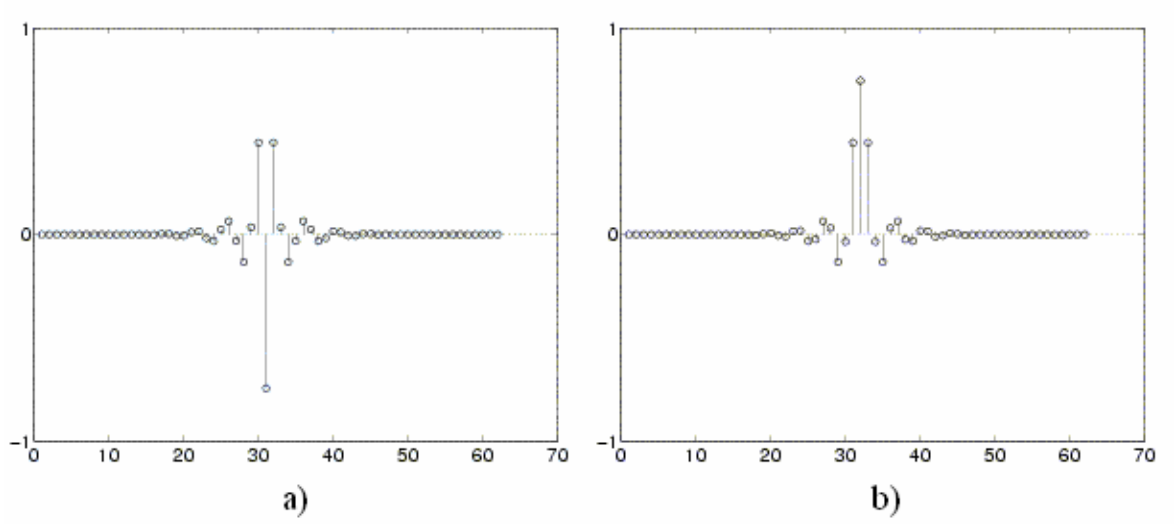
### *3.2.1.4 Meyer wavelet*

Tento mateřský wavelet je definován pouze ve frekvenční doméně a nemá explicitní vzorec pro časovou oblast. Je symetrický a ortogonální, lze ho tedy použít pro CWT i DWT.




Obr. 6. Meyer wavelet

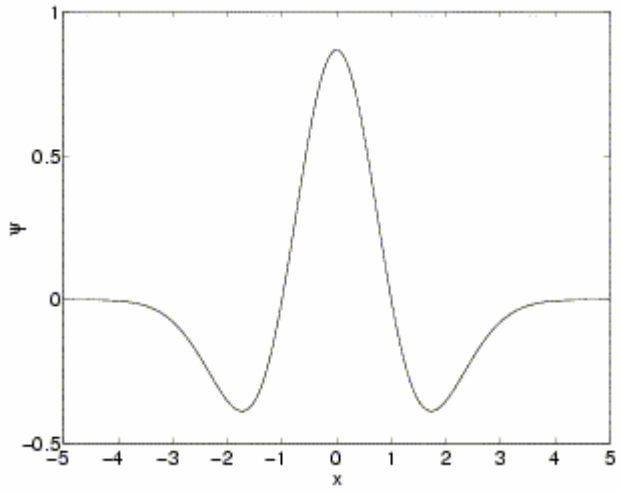
V originálním tvaru, v jakém je uveden na Obr.6. ovšem nejde realizovat pomocí FIR filtrů pro DWT a proto se vytváří jeho diskrétní aproximace pro horní a dolní propust uvedené na Obr.7.




Obr. 7. Diskrétní aproximace a) horní propusti a b) dolní propusti pro Meyer wavelet

### *3.2.1.5 Mexican hat*

Tento wavelet je členem skupiny Gaussovských waveletů tvořených jednotlivými derivacemi průběhu hustoty pravděpodobnosti Gaussova rozdělení. Má tvar druhé derivace Gaussovy křivky. Je symetrická, ovšem neortogonální, tudíž se hodí pouze pro CWT.




Obr. 8. Mexican hat

$$\psi(x) = \frac{2}{\sqrt{3}} \pi^{[1/4]} (1 - 2x^2) e^{-x^2} \tag{3.5}$$

## 3.3 DWT

Spojitá waveletová transformace je kvůli několika vlastnostem velmi obtížně použitelná v praxi. Podíváme-li se na rovnici (3.1), jež popisuje CWT, je potom jasné, že výpočet CWT je velice redundantní, jelikož wavelet je spojitě posouván po analyzovaném signálu s tím, že změna měřítka probíhá také spojitě. Výstupem transformace tedy bude nekonečný počet waveletových koeficientů, které jsou pro použití v praxi nadbytečné.

Neredundantní dekompozici signálu zajistíme vhodnou závislostí parametrů $s$ a $\tau$, viz. (3.6). Touto závislostí vytvoříme z waveletu ortonormální bázi.

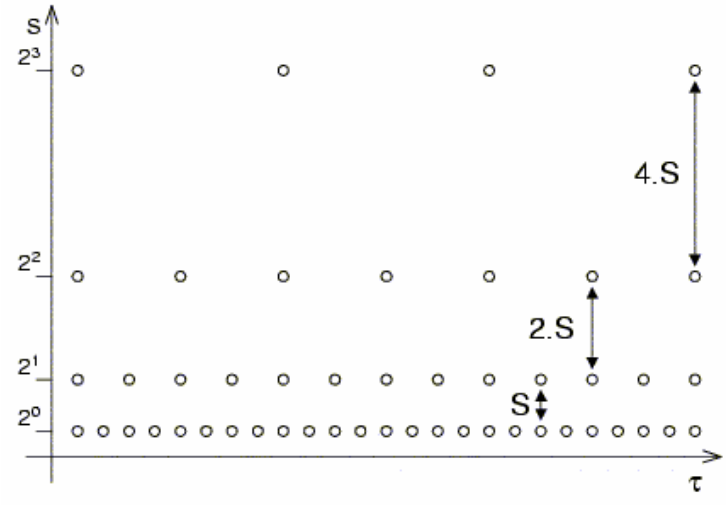
$$\begin{aligned} &s = 2^p, \tau = 2^p \cdot k \\ &p, k \in Z \end{aligned} \tag{3.6}$$

Parametr $p$ odpovídá měřítku, $k$ poloze.

Diskrétní wavelet pak vypadá dle (3.7).

$$\psi_{p,k}(t) = \frac{1}{\sqrt{|2^p|}} \psi\left(\frac{t - 2^p \cdot k}{2^p}\right) \tag{3.7}$$

Vzorkování prostoru $s, \tau$ tedy probíhá na tzv. dyadické mřížce, jež je ke shlédnutí na Obr.9.

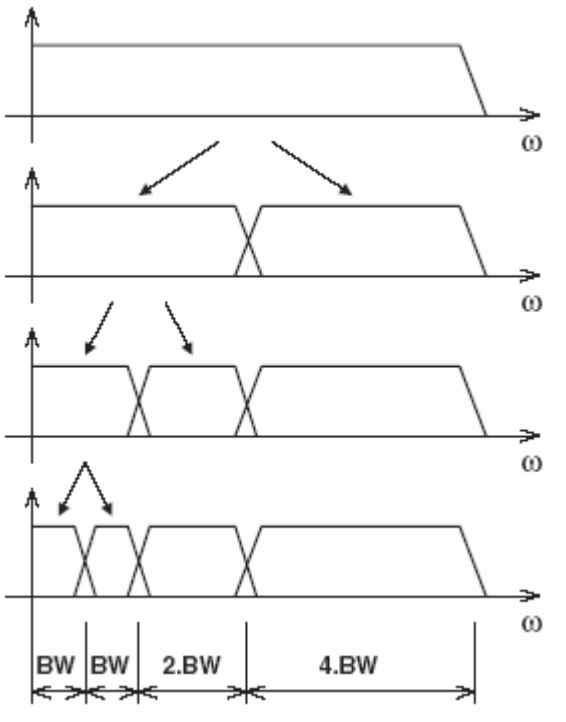



Obr. 9. Dyadická mřížka

Nejpoužívanější verzí je ekvidistantní vzorkování v čase $\tau$ a logaritmické vzorkování v měřítku $s$, protože umožňuje nestejnoměrně vzorkovat signál s ohledem na jeho časový nebo frekvenční průběh.

Díky takto zavedené ortonormalitě umožňuje wavelet neredundantní dekompozici signálu, tzv. analýzu s mnoha rozlišeními.

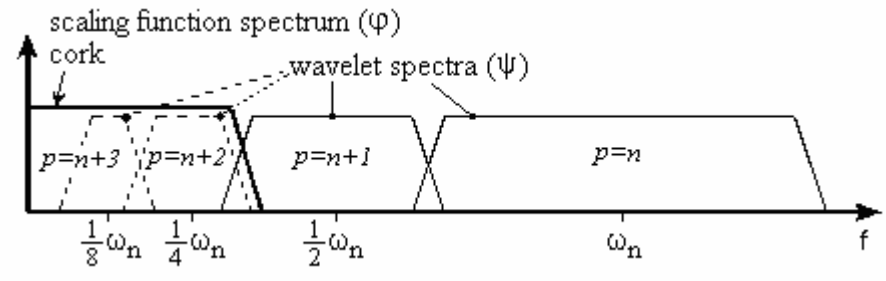
Diskrétní wavelet $\psi$ se chová jako horní pásmová propust filtrující signál kolem centrálního kmitočtu, který je závislý na měřítku mocninou dvou. V následujícím měřítku je filtrována horní polovina pásma předchozí dolnofrekvenční části signálu. S rostoucím kmitočtem roste šířka pásma (BW = *band width*) tohoto filtru. [1]
Postup je názorně zobrazen na Obr.10.




Obr. 10. Frekvenční pohled na DWT

Ovšem vyvstává zde problém, jak pokrýt spektrum úplně až k nule, protože vždy, když zkrátíme wavelet v čase o polovinu, šířka pásma waveletu bude také zkrácena o polovinu. Tedy s každým zkrácením o polovinu pokryjeme pouze další polovinu zbývajícího spektra. Tímto způsobem by bylo potřeba opět nekonečného počtu waveletů. Naštěstí existuje jednoduché řešení: spektrum nepokryjeme až k nule spektry waveletů, ale použijeme tzv. „zátku“ (cork). Tato je v podstatě dolní propustí (DP) a nazýváme ji *měřítkovou funkcí* (scaling function). [3]

Obr.11. ukazuje, jak je spektrum signálu pokryto wavelety a měřítkovou funkcí.




Obr. 11. Zavedení měřítkové funkce

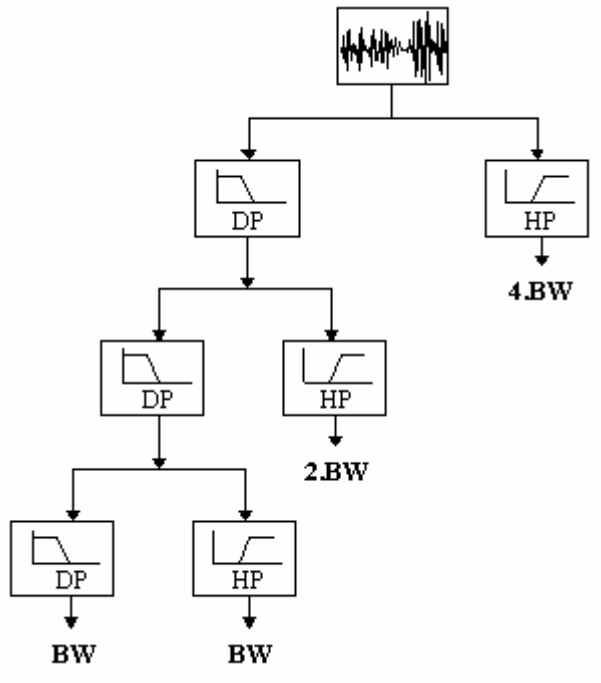
Podíváme-li se na tuto měřítkovou funkci jako na signál obsahující nízké frekvence, můžeme jej rozložit jako:

$$\varphi_{p,k}(t) = \sum_{p,k} W(p,k) \cdot \psi_{p,k}(t) \tag{3.8}$$

Volíme tedy funkci tak, aby její spektrum vyplnilo prostor nepokrytý spektry waveletů. Rovnice (3.8) využívá nekonečný počet waveletů až do určité hodnoty měřítka $p$. To znamená, že analyzujeme signál s využitím kombinace měřítkové funkce a waveletů, přičemž zátka překrývá i spektrum všech waveletů až do hodnoty měřítka $p$, zatímco zbytek pokryjí samotné wavelety. Tímto způsobem jsme omezili počet potřebných waveletů na konečný. [3]

Pokud tedy wavelet $\psi$ chápeme jako horní pásmovou propust a měřítkovou funkci $\varphi$ jako dolní propust, pak řadu dilatovaných waveletů můžeme společně s měřítkovou funkcí považovat za banku FIR filtrů, tedy filtrů s konečnou impulsní odezvou, a samotnou waveletovou transformaci za průchod signálu touto bankou.

Jak ukazuje Obr. 10. spektrum signálu je rozděleno na dvě shodné části - dolnopropustní (DP) a hornopropustní (HP). HP část obsahuje nejmenší detaily, které nás mohou zajímat. Jelikož DP část také obsahuje podrobnosti, je tato část rozdělena znovu a znovu, dokud není vytvořen žádaný počet subpásem. V tomto případě hovoříme o *iterační bance filtrů*. Výhodou této metody je, že je zapotřebí pouze dvou filtrů – filtru typu horní propust a dolní propust. Nevýhodou je fixní pokrytí signálního spektra. Tento druh analýzy se nazývá *rychlá waveletová transformace*. [3]
Její schéma je názorně zobrazeno na Obr.12.




Obr. 12. Aplikace banky filtrů

### 3.3.1 Rychlá DWT

Rychlá diskrétní waveletová transformace (FDWT) nabízí kompromis mezi efektivitou transformace a početní náročností potřebné pro analýzu signálu.

Jak již bylo řečeno, časově-frekvenční reprezentace digitálního signálu se získá použitím banky filtrů. Je tedy potřeba si tyto filtry definovat.

Filtr realizující dolní propust se nazývá *scaling filter* a je definován koeficienty:

$$S = (h_0, h_1, h_2, h_3, \ldots) \tag{3.9}$$

Filtr typu horní propust se nazývá *wavelet filter* a je definován koeficienty:

$$W = (g_0, g_1, g_2, g_3, \ldots) \tag{3.10}$$

#### *3.3.1.1 Přímá waveletová transformace*

Pomocí vektorů $S$ a $W$ lze definovat matici $A$, která bude využívána pro přímou waveletovou transformaci signálu:

$$A=\begin{pmatrix} h_0 & h_1 & h_2 & h_3 & . & . & . & . & . & . \\ g_0 & g_1 & g_2 & g_3 & . & . & . & . & . & . \\ . & . & h_0 & h_1 & h_2 & h_3 & . & . & . & . \\ . & . & g_0 & g_1 & g_2 & g_3 & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & h_0 & h_1 & h_2 & h_3 \\ . & . & . & . & . & . & g_0 & g_1 & g_2 & g_3 \\ h_2 & h_3 & . & . & . & . & . & . & h_0 & h_1 \\ g_2 & g_3 & . & . & . & . & . & . & g_0 & g_1 \end{pmatrix} \tag{3.11}$$

Matici $A$ lze aplikovat na hodnoty vektoru dat $x_n$ analyzovaného signálu:

$$A\cdot\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ x_N \end{pmatrix}=\begin{pmatrix} c_1 \\ d_1 \\ c_2 \\ d_2 \\ . \\ . \end{pmatrix} \tag{3.12}$$

Rozměr matice $A$ závisí na sloupcovém vektoru dat $x_n$, což vyplývá ze vzájemného vztahu mezi vektorem dat a filtry $h_n$, $g_n$.

Výsledkem prvního kroku přímé transformace jsou tedy diskrétní aproximace $c_n$ a koeficienty $d_n$.

#### *3.3.1.2 Zpětná waveletová transformace*

Aby byla waveletová transformace užitečná, musí existovat wavelet pro zpětnou transformaci. Zpětná transformace lze formulovat následovně:

$$X \cdot A \cdot B = X \tag{3.13}$$

kde $X$ je analyzovaný signál, $A$ je již výše zmíněná matice pro přímou waveletovou transformaci a $B$ je matice pro zpětnou waveletovou matici, kterou je nutné odvodit.

Aby byla rovnice (3.13) splněna, musí platit:

$$A \cdot B = I \tag{3.14}$$

tedy výsledkem součinu matic musí být jednotková matice. Toto zajistíme velmi jednoduše, pokud využijeme vlastnosti ortogonality, kdy:

$$A \cdot A^T = I \tag{3.15}$$

kde $A^T$ je matice transponovaná vzhledem k matici $A$.

Matice $B$ pro zpětnou waveletovou transformaci je tedy ortogonální k matici $A$:

$$B = \begin{pmatrix} h_0 & g_0 & . & . & . & . & . & . & h_2 & g_2 \\ h_1 & g_1 & . & . & . & . & . & . & h_3 & g_3 \\ h_2 & g_2 & h_0 & g_0 & . & . & . & . & . & . \\ h_3 & g_4 & h_1 & g_1 & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & h_2 & g_2 & h_0 & g_0 & . & . \\ . & . & . & . & h_3 & g_3 & h_1 & g_1 & . & . \\ . & . & . & . & . & . & h_2 & g_2 & h_0 & g_0 \\ . & . & . & . & . & . & h_3 & g_3 & h_1 & g_1 \end{pmatrix} \tag{3.16}$$

Zpětnou waveletovou transformací se tedy rozumí aplikace matice $B$ na vektor koeficientů $c_n$ a $d_n$:

$$B \cdot \begin{pmatrix} c_1 \\ d_1 \\ c_2 \\ d_2 \\ . \\ . \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ x_N \end{pmatrix} \tag{3.17}$$

Výsledkem bude rekonstrukce vektoru dat $x_n$ analyzovaného signálu $X$ .

### *3.3.1.3 Dekompozice signálu*

Po provedení přímé waveletové transformace získáme vektor koeficientů aproximace signálu $c_n$ a waveletové koeficienty $d_n$. Chceme-li provést několika-úrovňovou dekompozici signálu, je zapotřebí provést ještě jednu operaci a tou je podvzorkování $\downarrow 2$. Podvzorkování následuje vždy po kroku přímé waveletové transformace a má za úkol vybrat do dalšího kroku pouze koeficienty aproximace. Pro vektor dat velký $N = 8$ lze celý proces tedy zapsat následovně:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{pmatrix} \xrightarrow{transformace} \begin{pmatrix} c_1 \\ d_1 \\ c_2 \\ d_2 \\ c_3 \\ d_3 \\ c_4 \\ d_4 \end{pmatrix} \xrightarrow{\downarrow 2} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \end{pmatrix} \xrightarrow{transformace} \begin{pmatrix} C_1 \\ D_1 \\ C_2 \\ D_2 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \end{pmatrix} \xrightarrow{\downarrow 2} \begin{pmatrix} C_1 \\ C_2 \\ D_1 \\ D_2 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \end{pmatrix} \tag{3.18}$$

Výsledný vektor obsahuje koeficienty aproximace $C_n$ a waveletové koeficienty $D_n$ pro nejnižší rozlišení a dále waveletové koeficienty $d_n$ pro nejvyšší rozlišení. Pokud bude délka vektoru $N > 8$, bude samozřejmě zapotřebí většího počtu aplikací vztahu (3.18).

Diskrétní wavelet rekonstrukce je počítána inverzní metodou, počínaje úrovní nejnižšího rozlišení. [7]

## 3.4 Praktické užití wavetové transformace

Oblasti, ve kterých waveletová transformace dosahuje výborných výsledků je mnoho. Namátkou jsou uvedeny základní oblasti, ve kterých se waveletová transformace již uchytila a zefektivnila jak postupy, tak i výsledky předkládaných problémů.

1. **Pro potřeby experimentálního zpracování** – od té doby, co se wavelety ukázaly jako mechanismus pro experimentální zpracování dat, je jejich využití velmi atraktivní. Waveletová transformace poskytuje vizuální a informativní popis experimentálních výsledků. Umí vyčistit vstupní data od šumů a náhodných deformací a nastiňuje další způsoby pro zpracování a analýzu dat. Wavelety jsou výborné pro analýzu transientních signálů, které lze nalézt v medicíně, k analýzám potřebných pro kapitálový trh a jiné oblasti.

2. **Zpracování obrazu** – charakteristickým rysem lidského zraku je skutečnost, že se zaměřuje hlavně na významné detaily obrazu a nepotřené informace nebere v potaz. S waveletovou transformací můžeme vyhladit nebo zdůraznit některé detaily obrazu, přiblížit nebo oddálit určitou část (zoom), vybrat důležité detaily a dokonce zlepšit kvalitu celého obrazu.

3. **Komprese dat** – charakteristickou vlastností ortogonální analýzy je, že pro dostatečně hladká data jsou hodnoty detailů získané po transformaci velmi blízké nule. Proto jsou poté velmi jednoduše komprimovány běžnými statistickými metodami. Za velkou výhodu waveletové transformace se dá i považovat, že do vstupního signálu nejsou zaváděny žádné pomocné redundantní informace, ale signál může být plně rekonstruován použitím stejných filtrů. Kromě toho, po provedení transformace jsou detaily odděleny od hlavního signálu, čímž se velmi jednoduše implementuje ztrátová komprese. Pomocí waveletové transformace

mohou být signály komprimovány od 1/3 až do 1/10 bez ztráty významné informace (a více než 1/300 s tolerující ztrátou). Jako názorným příkladem užití waveletové transformace v kompresi dat může být standard MPEG-4.

4. **Neuronové sítě a další nástroje pro analýzu dat** – při trénování neuronové sítě (nebo nastavování parametrů jiných analytických nástrojů) představují šumy nebo velké množství náhodných špiček či průrev nebo harmonická zkreslení závažné problémy. Je tedy doporučeno vyčistit data před jejich analýzou. Vzhledem k výše zmíněným důvodům a díky rychlým a efektivním algoritmům se wavelety ukázaly být velmi vhodným a slibným nástrojem pro čištění a předzpracování dat, která jsou používána ve statistických a obchodních aplikacích, systémech umělé inteligence atd.

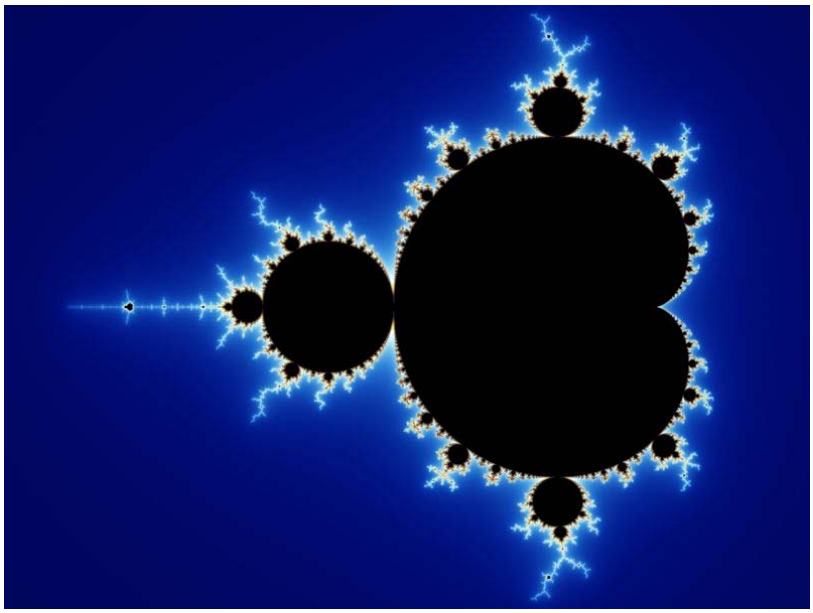
5. **Systémy přenosu dat a digitálního zpracování signálů** – díky vysoké výkonnosti algoritmů a jejich stabilitě proti šumu může být waveletová transformace mocným nástrojem v oblastech, kde bývaly ostatní metody analýzy dat běžně používány (jako Fourierova transformace). Díky zvláštním schopnostem a kvalitám waveletové transformace v časově-frekvenční oblasti můžou být systémy pro přenos a zpracování dat významně rozšířeny a zefektivněny. [4]

# 4 FRAKTÁLNÍ KOMPRESE

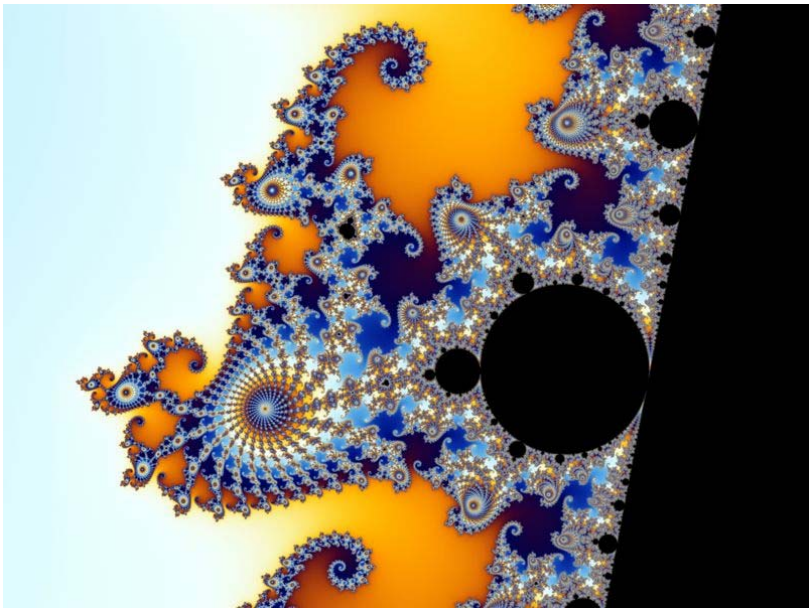
Fraktální komprese je matematický postup používaný ke kódování bitmapových obrazů reálného světa jako množiny matematických dat, která vyjadřují fraktální vlastnosti obrazu. Jde o ztrátovou metodu a vychází ze skutečnosti, že všechny přirozené a většina umělých obrazů obsahují nadbytečné informace ve formě podobných, opakujících se vzorů, tzv. fraktálů.

## 4.1 Fraktální geometrie

Fraktální geometrie je samostatný a již poměrně rozsáhlý vědní obor, který zasahuje i do mnoha dalších disciplín. Intenzivně je rozvíjena již zhruba od šedesátých let 20. století a za zakladatele fraktální geometrie je považován vědec polského původu Benoit B. Mandelbrot. Ten jako první matematicky definoval pojem *fraktál* (z latinského *fractus* = zlomit). O definici fraktálu se vědci pokoušeli už před Mandelbrotem, ovšem jejich popis byl velmi neúplný a proto je Mandelbrotovi v této záležitosti připisováno prvenství. Pravděpodobně nejznámějším spojením Mandelbrotova jména s fraktální geometrií je právě jeden významný fraktál – Mandebrotova množina. Takto byl Johnem Hubbardem pojmenován dynamický fraktál ležící v komplexní rovině. Ve skutečnosti tedy Mandelbrot nebyl první, kdo tento fraktál pomocí výpočetní techniky vykreslil, jako první ho však v roce 1979 prezentoval. [5]



Obr. 13. Obarvená Mandelbrotova množina



Obr. 14. Část Mandelbrotovy množiny

### 4.1.1 Topologická dimenze

Geometricky hladké objekty, které je možné popsat klasickou Euklidovskou geometrií, mají celočíselnou dimenzi, která se také velmi často nazývá *dimenzí topologickou*. Před tím, než si vědci všimli, že existují fraktální útvary, si vystačili právě s touto jednoduchou dimenzí. Když se na celou problematiku podíváme z jednoduššího hlediska, můžeme říci, že topologická dimenze určuje počet parametrů (nezávislých proměnných), kterým lze dané těleso (resp. každý bod na tělese) popsat. Například bod má nulovou dimenzi, protože je sám popsán vztahem $P = X$ (tj. konstantním vektorem). Naproti tomu má úsečka dimenzi rovnu jedné, protože ji můžeme popsat vztahem $y_t = y_0 + k \cdot t$, kde $t$ je jediný parametr. Pozici každého bodu, který leží na této úsečce tedy lze vyjádřit výše uvedeným vztahem.

To, že má nějaká křivka dimenzi rovnu jedné, ovšem ještě nutně neznamená, že je zobrazována pouze v jednorozměrném prostoru. Dimenze udává jen počet parametrů, které jsou nutné pro jednoznačné definování bodu na křivce. Například následující křivka má

dimenzi rovnu jedné, ale je zobrazována v trojrozměrném prostoru: $x = \sin(t) \cdot \log(t), y = \cos^2(t), z = t$

Jediným použitým parametrem je zde opět parametr $t$, ovšem poloha bodu je určena trojicí souřadnic $x$, $y$ a $z$. [5]

### 4.1.2 Hausdorffova dimenze

Objekty popsané fraktální geometrií mají dimenzi neceločíselnou. Tuto dimenzi nazýváme fraktální nebo také Hausdorffovou dimenzí a její hodnota (resp. míra rozdílu mezi fraktální dimenzí a dimenzí topologickou) pak udává úroveň členitosti daného fraktálního objektu.

Pokud měříme délku geometricky hladké křivky, která má topologickou dimenzi rovnu jedné, dostaneme při jejím pohledu v různých měřítkách vždy stejné číslo. Měřením délky břehu ostrova (což se dá považovat za křivku s topologickou dimenzí rovnou jedné) se při zmenšování měřítka toto číslo stává nekonečně velkým. Pobřeží tedy v rovině zabírá více místa než hladká křivka. Nezabírá však všechno místo, tedy nevyplňuje celou rovinu. Jeho skutečná dimenze je tedy větší než topologická dimenze křivky (ta je rovna jedné) a současně je menší než topologická dimenze roviny (ta je rovna dvěma). Z toho tedy vyplývá, že dimenze tohoto útvaru není celočíselná. A právě toto neceločíselné číslo se nazývá Hausdorffovou dimenzí.

Hodnota Hausdorffovy dimenze udává, s jakou rychlostí délka těchto útvarů (nebo odpovídající veličina při větším počtu rozměrů, tj. povrch nebo objem) roste do nekonečna. Pokud se bude Hausdorffova dimenze a topologická dimenze lišit velmi málo, bude takový objekt málo členitý. Bude-li Hausdorffova dimenze ostře větší než dimenze topologická, bude objekt velmi členitý. Mezní hodnotou je případ, kdy je Hausdorffova dimenze o jedničku větší než dimenze topologická - tuto vlastnost má například hranice Mandelbrotovy množiny. [5]

### 4.1.3 Fraktál

Rozdíl mezi topologickou dimenzí a Hausdorffovou dimenzí použil Mandelbrot při definici fraktálu následovně:

*"Fraktál je množina či geometrický útvar, jejíž Hausdorffova dimenze je (ostře) větší než dimenze topologická".*

Ne všichni matematici ovšem tuto definici uznávají, protože ve skutečnosti žádná matematicky přesná definice fraktálu neexistuje. [5]

### 4.1.4 IFS

Fraktály lze na počítači generovat několika osvědčenými způsoby, z nichž nejvýznamnějším pro kompresi obrazu jsou *Iterated Function Systems* (IFS) - systémy iterovaných funkcí. První publikace, které se týkaly IFS systémů, vydali v roce 1985 Stephen Demko a následně pak v roce 1987 Michael Barnsley. Tvorba obrázků pomocí IFS systémů patří mezi generativní metody vytváření fraktálů, kterou řadíme mezi metody deterministické.

IFS je soubor parametrů $w_i$ , které definují afinní transformace. Afinní transformace mají tu vlastnost, že jsou lineární, to znamená, že při aplikaci této transformace na úsečku, resp. množinu bodů ležících na přímce, bude výsledkem transformace rovněž úsečka (resp. množina bodů, které leží na přímce). Množina těchto transformací pak určuje výsledný fraktál. Aby bylo zajištěno, že po aplikaci transformací bude obrázek konvergovat do chtěného atraktoru (všechny body budou aplikací transformací putovat do jednoho, tzv. pevného bodu), musí být transformace kontraktivní (zmenšující).

Pro každý bod při aplikaci afinní transformace platí tato rovnice:

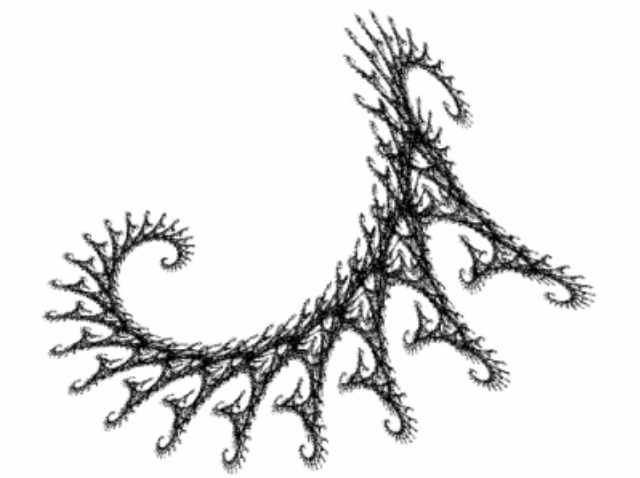
$$w_i \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \end{bmatrix} \tag{4.1}$$

Koeficienty afinní transformace jsou definovány jako:

$$\begin{aligned} a &= r_1 \cdot \cos\theta \\ b &= -r_2 \cdot \sin\delta \\ c &= r_1 \cdot \sin\theta \\ d &= r_2 \cdot \cos\delta \end{aligned} \tag{4.2}$$

kde $r_1$ , $r_2$ jsou kontraktivní faktory a $\theta$ , $\delta$ jsou rotace kolem os. Zbývající parametry $e$ , $f$ určují posunutí v souřadném systému ze staré pozice $x', y'$ na novou pozici $x, y$ .

Výsledná množina bodů, které tvoří fraktál, je tedy tvořena iterativním prováděním operací zmenšování, rotace a posuvu nově získaných útvarů.



Obr. 15. Útvar vytvořený pomocí IFS

## 4.2 Principy fraktální komprese

Po zavedení matematického popisu IFS, pomocí kterého lze vytvořit přirozeně vypadající obraz, vyvstala otázka, zda tento popis není možné využít v opačném směru – tedy k popisu již existujícího obrazu a k jeho kompresi. Popis obrazu pomocí IFS je znám jako *inverzní problém*. Tento problém není ještě v současné době plně vyřešen.

Michal Barnsley, který je průkopníkem IFS, založil v roce 1987 společnost „*Iterated Systems Incorporated*" a nechal si patentovat vlastní postup na řešení inverzního problému. Zároveň publikoval své výsledky a uvedl i několik případů obrazů zkomprimovaných pomocí IFS, u nichž dosáhl kompresního poměru až 10 000:1. Ovšem tato komprese obrazu byla velmi náročná – trvala okolo 100 hodin na jednom z nejrychlejších počítačů té doby a to dokonce s nezbytnou asistencí operátora.

V roce 1988 Barnsleyho student Arnauld Jacquin modifikoval IFS tak, že se již v obrazu nehledaly věrné kopie obrazu celého, ale pouze jeho částí. Tento nový postup, který si nechal Barnsley také patentovat, byl pojmenován *Partitioned IFS* (PIFS) a obrovským přínosem byla plná automatizace kompresního algoritmu ovšem na úkor kompresního poměru. Ten se pohyboval od 8:1 do 50:1. I přes menší kompresní poměr jsou všechny verze fraktálních kompresních algoritmů založeny na PIFS. [6]

PIFS tedy vyhledává podobnost mezi částmi obrazu, přičemž každá část má kromě své polohy dané souřadnicemi $x, y$ také úroveň intenzity, kterou můžeme interpretovat hodnotou na ose $z$. Do původní IFS se tedy musí započítat třetí dimenze, takže v konečném efektu budou $z$ -ovou souřadnici každého bodu transformovat dvě hodnoty. Je to posun $o_i$ (offset) a škálovací faktor $s_i$ (scale) pro úpravu kontrastu.

$$w_i \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \\ o_i \end{bmatrix} \tag{4.3}$$

přičemž transformace $w_i$ zůstává kontraktivní ve všech 3 směrech: $x, y, z$

Hledaná transformace signálu $f$ je tedy sjednocením afinních transformací všech částí tohoto obrazu. Fraktální komprese je ztrátová metoda, což můžeme pro signál $f$ daný aproximací $W$ zapsat jako:

$$f \approx W(f) = \bigcup_{i=1}^{n} w_i(f) \tag{4.4}$$

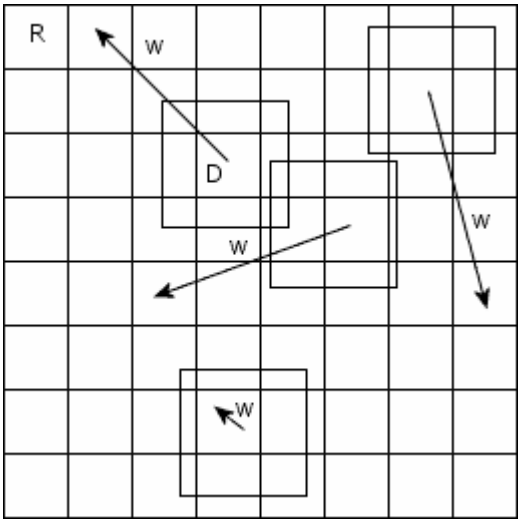
### 4.2.1 Kódování

Cílem komprese je tedy snaha o nalezení takového signálu $g$ (resp. jeho transformaci), který je velmi blízký původnímu signálu $f$ :

$$f \approx g = W(g) \approx W(f) \tag{4.5}$$

Signál $f$ se rozdělí na pravidelné *oblastní bloky* $R_i$ (range block), které se nepřekrývají a pokrývají celou jeho plochu. Jejich sjednocením je tedy celý signál $f$. Ke každému z těchto oblastní bloků je dohledáván nejpodobnější *doménový blok* $D_i$ (domain block). Doménové bloky se mohou překrývat a nemusí dohromady pokrývat celou plochu signálu $f$. Přitom ovšem musí být větší než oblastní bloky, aby byla zajištěna kontraktivita.

Na Obr.16. je ukázáno hledání příslušného doménového bloku k bloku oblastnímu. Principem komprese je nalézt takový $D_i$ a jeho transformaci $w_i$ tak, že po aplikaci $w_i$ na oblast signálu určenou $D_i$ získáme velmi podobnou část signálu jako $R_i$.




Obr. 16. Mapování doménových bloků

Cílem problému je minimalizovat vzdálenost hodnot mezi $R_i$ a $D_i$.

Vycházíme-li z mapování doménového bloku $D_i$ na oblastní blok $R_i$ podle rovnice:

$$R_i \approx s_i \cdot D_i + o_i \tag{4.6}$$

pak k nalezení optimálních hodnot $s_i$ a $o_i$ použijeme kvadratickou Euklidovskou vzdálenost:

$$E = \sum_{i=1}^{n} \left[ \left( s \cdot d_i + o \right) - r_i \right]^2 \tag{4.7}$$

kde $d_i$ jsou jednotlivé hodnoty doménového bloku $D_i$ a $r_i$ jsou hodnoty oblastního bloku $R_i$.

Parciální derivace rovnice (4.7) podle $s$ a $o$ položíme rovny 0:

$$\frac{\delta E}{\delta s} = 0, \frac{\delta E}{\delta o} = 0 \tag{4.8}$$

což vede ke vztahům pro výpočet parametrů $s$ a $o$ :

$$s = \frac{\left[ n^2 \cdot \left( \sum_{i=1}^{n} d_i r_i \right) - \left( \sum_{i=1}^{n} d_i \right) \cdot \left( \sum_{i=1}^{n} r_i \right) \right]}{\left[ n^2 \cdot \sum_{i=1}^{n} d_i^2 - \left( \sum_{i=1}^{n} d_i \right)^2 \right]} \tag{4.9}$$

$$o = \frac{\left[ \sum_{i=1}^{n} r_i - s \cdot \sum_{i=1}^{n} d_i \right]}{n^2} \tag{4.10}$$

Výpočtem škálovacího faktoru a posunu a s geometrickými informacemi o transformaci jsme získali předpis pro mapování doménového bloku $D_i$ na oblastní blok $R_i$. Tyto výpočty se provádějí pro všechny oblastní bloky, které pokryjí celou plochu analyzovaného signálu. Získáme tak soubor transformací $w_i$, které splňují vztah (4.4).

### 4.2.2 Dekódování

Dekódování spočívá v iterativní aplikaci získaných transformací $W(f)$ na oblast nulového signálu $g$. Nulový signál $g$ je znovu rozdělen na oblastní bloky $R_i$ a na každý $R_i$ aplikujeme příslušnou transformaci $w_i$. To má za následek načtení doménového bloku $D_i$ na příslušných souřadnicích, provedení jeho geometrické transformace (rotace, překlopení, kontrakce) a každá hodnota doménového bloku je vynásobena škálovacím faktorem $s_i$ a přičten posun $o_i$. Tyto operace se provádějí

iterativně. Při každé iteraci se totiž zvýší rozlišení dekódovaného signálu. Zmenšený doménový blok se stane součástí jiného doménového bloku, který se při jiné transformaci zase mapuje, čímž se zvyšuje detail.

Jak ukazuje rovnice (4.6), po několika iteracích tak získáme oblastní blok $R_i$, který je hodnotově velmi podobný, jako při kódování v signálu $f$.

### 4.2.3 Urychlení kompresního algoritmu

Vyhledávání nejpodobnějšího doménového bloku ke každému oblastnímu bloku postupným vzájemným porovnáváním je nejnáročnější částí algoritmu. Z tohoto důvodu vzniklo mnoho variant kompresního algoritmu, které se liší způsobem, jakým omezují počet prováděných porovnání a tím i urychlují kompresi:

1. **Těžká hrubá síla** – Tato metoda nijak neomezuje počet porovnání a pro každý blok prohledává doménové bloky na všech pozicích. Podává sice nejlepší výsledky, ale pro svou výpočetní náročnost se nepoužívá.
2. **Lehká hrubá síla** – Nejjednodušší způsob, jak omezit množství porovnání, je omezení možných poloh doménových bloků. Při porovnání se například uvažují jen bloky na sudých souřadnicích, což omezí počet srovnání na čtvrtinu oproti metodě „těžké hrubé síly“. Omezení může samozřejmě být i rozsáhlejší.
3. **Náhodná volba domén** – V tomto případě se pozice doménových bloků generují pomocí pseudonáhodných funkcí. Jako řídicí parametr se zpravidla dává počet těchto generací (resp. počet porovnání doménového bloku na dané pozici s oblastním blokem).
4. **Omezení prohledávané oblasti** – Při hledání podobného doménového bloku se neprohledává celý obraz, ale jen určitá omezená plocha. Například jen okolí oblastního bloku do určité vzdálenosti. Více sofistikované alternativy rozdělí obraz nejdříve na oblasti s podobnými vzory a následně prohledávají jen oblast, ve které leží daný oblastní blok.
5. **Spirála** – Vyhledávání do spirály je určitou modifikací předchozí metody. Prohledávání se začne na souřadnicích, kde leží oblastní blok. Dále se pokračuje

ve spirále. V tomto případě se nehledá nejlepší pár, ale hledaní skončí v okamžiku, kdy je nalezen doménový blok s dostačující podobností.

6. **Vyhledávání „na místě“** – Tato metoda je stejně jako metoda „těžká hrubá síla“ uváděna jen pro srovnání jako dolní mez dosažitelné doby vyhledávání párů. Jako párový doménový blok je vždy vybrán ten, který je na stejných souřadnicích jako oblastní blok.

7. **Kategorizace doménových bloků** – Do této kategorie spadá mnoho různých metod. Pro všechny obecně platí, že se před započetím hledání párů rozdělí všechny doménové bloky do několika kategorií, případně ohodnotí podle určitého kritéria na spojité škále. Každý oblastní blok se při prohledávání zařadí resp. ohodnotí stejným způsobem a dále se již porovnává pouze se stejně zařazenými doménovými bloky.

## 4.3 Vlastnosti fraktální komprese

Fraktální komprese má několik vlastností, které ji činí specifickou na rozdíl od ostatních kompresních metod. O fraktální kompresi můžeme říci, že je:

1. **Ztrátová** – Nehledá se věrná kopie obrazu (části obrazu), ale přibližná. Z toho plyne, že degradace obrazu, která vzniká, je jiného druhu než u běžných kompresních algoritmů. Díky fraktálním složkám obrazu se změny jeví mnohem přirozeněji. Důsledkem toho je velmi obtížné srovnávat úspěšnost fraktální komprese s jinými ztrátovými metodami. Dá se jen těžko porovnat kvalita obrazu zkomprimovaného při stejném kompresním poměru různými algoritmy, vzhledem k tomu že ztráta kvality se projeví odlišně.

2. **Asymetrická** – Kódování je časově velmi náročný proces a na rozdíl od něj je aproximace původního obrázku z transformací velmi rychlá. Můžeme tedy říci, že fraktální komprese je silně asymetrická metoda.

3. **Nezávislá na rozlišení** – Dekomprimovaný obraz má znaky fraktálu. Je možné ho zvětšovat prakticky donekonečna a stále se objevují nové detaily (ty nejsou samozřejmě obsaženy v původním komprimovaném obraze, ale jsou uměle dopočítány). Tato vlastnost, tedy nezávislost na rozlišení, se dá velmi využít při zoomování a zvětšování obrazů.

4. **Nezávislá na barevné hloubce obrazu** – Velikost dat po kompresi je stále stejná bez ohledu na počet bitů na pixel v komprimovaném obraze. Z toho plyne, že kompresní poměr se zlepšuje lineárně s počtem bitů na pixel.
5. **Patentovaná** – Na rozdíl od jiných kompresních algoritmů je fraktální komprese od roku 1991 patentovaná Michaelem Barnsleyem a tudíž pouze držitel tohoto patentu ji může využívat ke komerčním účelům.

# II. PRAKTICKÁ ČÁST

# 5 APLIKACE WAVELETOVÉ TRANSFORMACE

## 5.1 Volba vhodných bázových funkcí

Volba bázových funkcí (tedy mateřských waveletů) spočívá v určení koeficientů pro filtry dolní a horní propusti – tedy určení vektorů $S$ a $W$ (viz. (3.9) a (3.10)). Pro různé báze existuje různý počet koeficientů a různá vyjádření těchto koeficientů a právě ty jsou pro výběr bázové funkce pro danou úlohu zásadní. Často se v praxi používají wavelety, které podávají lepší výsledky bez ohledu na jejich náročnost, případně se volí taková bázová funkce, která je pro danou úlohu optimální z obou hledisek – jak výsledné kvality transformace, tak výpočetního výkonu potřebného k analýze signálu.

### 5.1.1 Koeficienty Haarovy báze

Nejjednodušší bázovou funkcí je Haar wavelet, který má nejvyšší zkreslení a tím také nejmenší účinnost při waveletové analýze. Jeho největší výhodou je ovšem nejnižší počet koeficientů. Při použití Harrova waveletu se pracuje se 2 koeficienty, takže dolnopropustní *scaling filter* $S$ vypadá následovně:

$$S = (h_0, h_1) \tag{5.1}$$

kde prvky vektoru jsou pevně dány

$$\begin{aligned} h_0 &= 1/\sqrt{2} = 0.7071067812 \\ h_1 &= 1/\sqrt{2} = 0.7071067812 \end{aligned} \tag{5.2}$$

přičemž platí, že

$$h_0^{\,2} + h_1^{\,2} = 1 \tag{5.3}$$

což dokazuje podmínku ortonormality bázové funkce.

*Wavelet filter* $W$, který je horní propustí, je definován jako

$$W = (g_0, g_1) \tag{5.4}$$

Koeficienty $W$ filtru získáme na základě teorie kvadraturního zrcadlového filtru jako doplněk k $S$ podle vztahu:

$$g_n = (-1)^n \cdot h_{N-1-n} \tag{5.5}$$

kde $N$ je počet koeficientů filtru. V případě Haarovy báze je tedy $N = 2$.

Koeficienty filtru horní propusti $W$ jsou tedy ze vztahu (5.5) vypočítány jako

$$\begin{aligned} g_0 &= h_1 \\ g_1 &= -h_0 \end{aligned} \tag{5.6}$$

přičemž platí, že bázová funkce musí mít nulovou střední hodnotu

$$g_0{}^2 + g_1{}^2 = 0 \tag{5.7}$$

### 5.1.2 Koeficienty báze Daubechies 2.řádu

Mnohem lepších výsledků při transformaci signálu se dosáhne použitím Daubechies báze, která je přímo optimalizována pro aproximaci konstantních a lineárních signálů a tedy je pro zpracování statických obrazů velmi vhodná.

Filtrů této báze existuje několik typů a liší se počtem a druhem koeficientů. V této práci byl použit Daubechies wavelet 2. řádu, který má dolní propust $S$ definovanou jako:

$$S = (h_0, h_1, h_2, h_3) \tag{5.8}$$

Používají se tedy 4 koeficienty, které mají toto vyjádření:

$$\begin{aligned} h_0 &= \frac{1}{4}(1+\sqrt{3})/\sqrt{2} = 0.4829629131445341 \\ h_1 &= \frac{1}{4}(3+\sqrt{3})/\sqrt{2} = 0.8365163037378077 \\ h_2 &= \frac{1}{4}(3-\sqrt{3})/\sqrt{2} = 0.2241438680420134 \\ h_3 &= \frac{1}{4}(1-\sqrt{3})/\sqrt{2} = -0.1294095225512603 \end{aligned} \tag{5.9}$$

*Wavelet filter* $W$ má podle vztahu (5.5) své koeficienty

$$\begin{aligned} g_0 &= h_3 \\ g_1 &= -h_2 \\ g_2 &= h_1 \\ g_3 &= -h_0 \end{aligned} \tag{5.10}$$

Přičemž platí ortonormalita

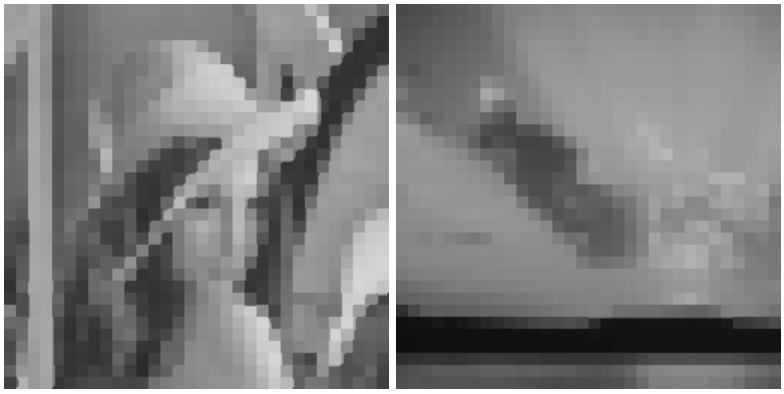
$$h_0^{\,2} + h_1^{\,2} + h_2^{\,2} + h_3^{\,2} = 1 \tag{5.11}$$

a také nulová střední hodnota bázové funkce

$$g_0^{\,2} + g_1^{\,2} + g_2^{\,2} + g_3^{\,2} = 0 \tag{5.12}$$

### 5.1.3 Rekonstrukce obrazu pomocí Haarovy báze a Daubechies 2. řádu

Pro názornou ukázku vlastností použitých bází jsou uvedeny příklady rekonstrukce obrazu. Nejprve byla provedena tří-úrovňová dyadická dekompozice obrazu pomocí waveletové transformace, waveletové koeficienty byly nulovány, takže zůstaly k dispozici pouze koeficienty aproximace. Z této aproximace byl obraz zpět rekonstruován. Na Obr.17. je zobrazena rekonstrukce obrazu, při jejíž analýze a syntéze byla použita Harrova báze. Na Obr.18. byla použita báze Daubechies.



Obr. 17. Rekonstrukce obrazu pomocí Harrovy báze

Obr. 18. Rekonstrukce obrazu pomocí Daubechies báze

Lze tedy vidět, že Daubechies umožňuje hladkou rekonstrukci signálu oproti Haarově bázi. Naproti tomu je implementace Harrovy báze o něco jednodušší a jeho výpočetní náročnost je nižší než v případě Daubechies.

### 5.1.4 Výpočetní náročnost

U waveletové transformace platí, že čím vyšší je počet koeficientů filtru, tím přesnější je úroveň aproximace analyzovaného signálu. Ovšem se zvyšujícím se počtem koeficientů narůstá i výpočetní náročnost a tím i doba provedení transformace.

Mějme vektor dat délky $L$ a počet koeficientů filtru $N$ . Wavelet transformace je lokální operací pro kterou platí $N << L$ a pro nejvyšší frekvenční pásmo vyžaduje první stupeň dekompozice $2NL$ násobků a sečtení. Pro další frekvenční pásmo je délka vektoru diskrétních aproximací $c_n$ redukována na $L/2$. Každý další stupeň dekompozice tedy vyžaduje $2(NL/2)$ úkonů. Celkový počet operací pro waveletovou dekompozici je

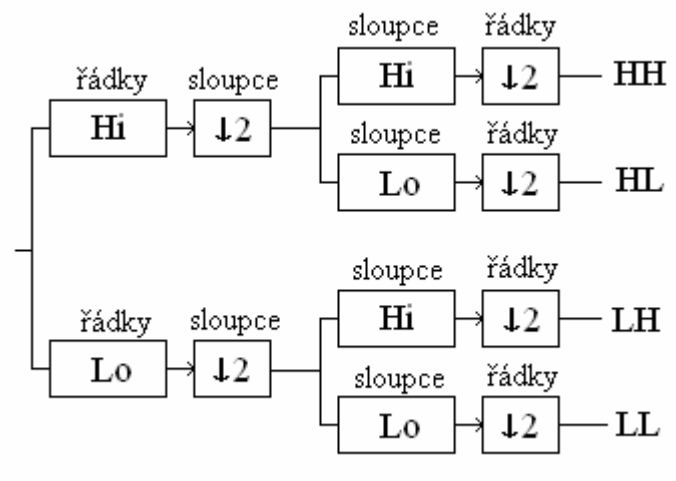
$$2\left( NL + \frac{NL}{2} + \frac{NL}{4} + \ldots \right) = 2NL\left( 1 + \frac{1}{2} + \frac{1}{4} + \ldots \right) \approx 4NL \qquad (5.13)$$

Ortonormální waveletová transformace vyžaduje pouze $O(L)$ výpočtů. Je tedy dokonce rychlejší než Rychlá Fourierova transformace (FFT), která vyžaduje $O(L \cdot \log_2 L)$ operací násobení a sčítání. [7]

## 5.2 Dekompozice obrazu

Při dekompozici obrazu se vstupní signál (tedy matice obrazových dat) nechá projít bankou filtrů typu horní propust k analýze vyšších frekvencí a bankou filtrů typu dolní propust k analýze nižších frekvencí. Signál se tedy dělí na aproximaci (reprezentován nižšími frekvencemi zpracovaného signálu) a detailnější informaci (reprezentován vyššími frekvencemi zpracovaného signálu). Potom následuje podvzorkování, pomocí kterého se odstraní část vzorků v signálu.

Na Obr.19. můžeme vidět základní strukturu realizující dyadický rozklad (dekompozici) obrazu. Bloky $Hi$ resp. $Lo$ představují impulsní odezvu filtru typu horní propust resp. dolní propust a blok s označením $\downarrow 2$ provádí podvzorkování.




Obr. 19. Základní struktura pro dyadický rozklad 2D signálu

Podle schématu tedy můžeme vidět, že se nejprve transformují všechny řádky obrazu (tedy pixely ve všech řádcích). Takto zanalyzovaná data se sloupcově podvzorkují a následně se transformují sloupce těchto dat. Po řádkovém podvzorkování je takto proveden dyadický rozklad první úrovně a ve výsledku jsou k dispozici 4 sady koeficientů:

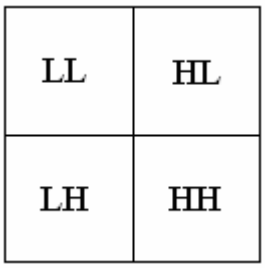
LL – aproximace obrazu

LH – detaily obrazu v horizontálním směru

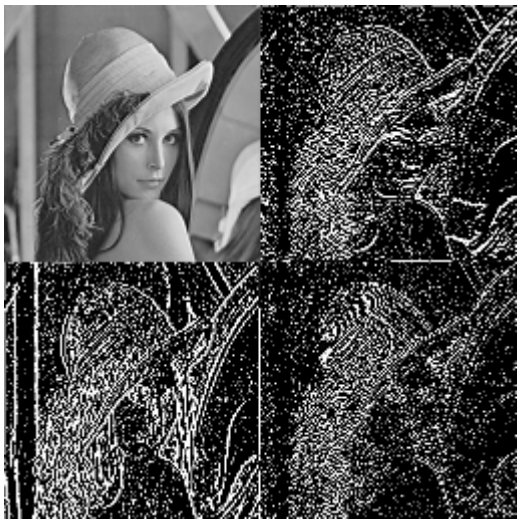
HL – detaily obrazu ve vertikálním směru

HH – detaily obrazu v diagonálním směru

Obr.20. a Obr.21. ukazují, jak jsou tyto 4 sady rozmístěny v matici koeficientů 2D signálu. Po provedení dyadického rozkladu první úrovně jsou nejvíce důležité informace (tedy samotná aproximace obrazu) umístěny v levé horní submatici signálu a nejvíce redundantní informace v pravé dolní submatici signálu.

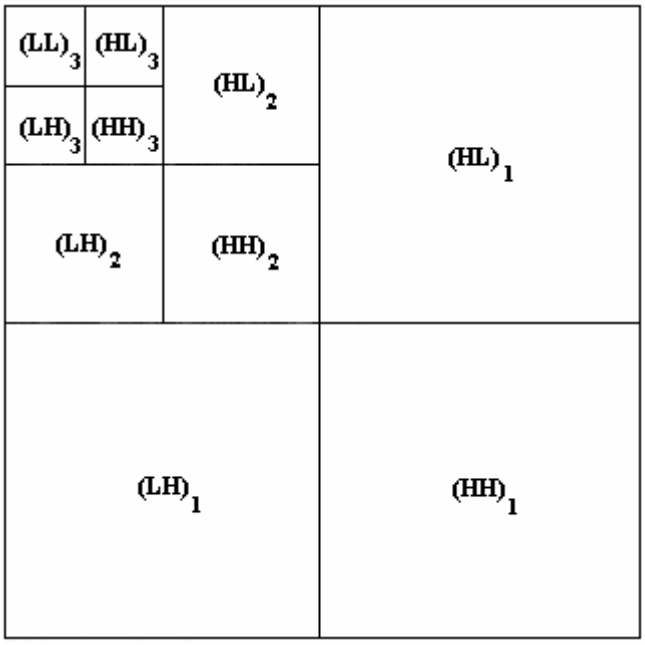



Obr. 20. Schéma dyadického rozkladu 2D signálu



Obr. 21. Dyadický rozklad první úrovně obrazu „Lena“

Pro mnoho-úrovňovou dekompozici signálu se často používá nestandardní dyadický rozklad, u kterého se vždy rekurzivně analyzuje pouze aproximační část (LL pásmo) z předchozího kroku dekompozice. Na Obr. 22. je schématicky zobrazena dekompozice obrazu třetí úrovně, která se často používá v oblasti zpracování statických obrazů.




Obr. 22. Schéma dyadického rozkladu třetí úrovně

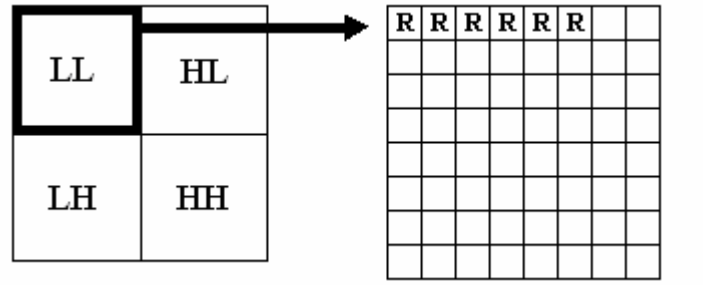
V této práci se ovšem používá dyadický rozklad obrazu pouze první úrovně. Účelem takto dekomponovaného obrazu je získat koeficienty aproximace, které jsou poté zpracovávány dalšími algoritmy zajišťující kompresi obrazu.

# 6 APLIKACE FRAKTÁLNÍ KOMPRESE

V teoretické části v kapitole 4.2. byly popsány principy fraktální komprese, které jsou využity i v této práci. Za signál $f$ je v tomto případě považováno LL pásmo dyadicky rozloženého obrazu – tedy matice koeficientů aproximace, které se získaly po waveletové analýze. Na tyto koeficienty jsou aplikovány algoritmy fraktální komprese.

## 6.1 Kódování

Všechny koeficienty obsažené v nízko-frekvenčním sub-pásmu jsou uloženy do nové pracovní matice datového typu *float* (koeficienty aproximace jsou totiž s plovoucí řádovou čárkou). Na této pracovní matici jsou prováděny veškeré kompresní algoritmy. Oblastní bloky (range blocks R), které se v pracovní matici nacházejí, jsou tedy tvořeny hodnotami koeficientů aproximace.




Obr. 23. Použití LL pásma jako výchozí matici pro kompresní algoritmy

Pracovní matice není přímo rozdělována na oblastní bloky, ale přistupuje na jejich koordináty pomocí jejich pořadového čísla a známé velikosti. Tím je zajištěn rychlý pohyb po pracovní matici.

Pro každý oblastní blok je tedy dohledáván nejpodobnější doménový blok (domain block D), který byl zmenšen na velikost oblastního bloku pomocí podvzorkování, čímž je zajištěna kontraktivita. Pozice doménových bloků, které se s oblastním blokem porovnávají, jsou generovány pseudonáhodně. Je tedy použita varianta **náhodné volby domén**. Počet doménových bloků, které jsou s jedním oblastním blokem porovnávány, je dán počtem generací jejich pozic. Čím větší počet generací bude zadáno, tím větší je šance,

že bude nalezen podobnější doménový blok. To je ovšem na úkor výpočetní náročnosti a tedy i doby, kterou komprese zabere. Navíc při každém porovnávání doménového bloku jsou aplikovány různé geometrické transformace, které jsou následně popsány v kapitole 6.1.1.

Symbolicky lze celý postup zapsat následovně:

```
Pro všechny R bloky ve sloupci
{
    Pro všechny R bloky v řádku
    {
        Pro zadaný počet pokusů o nalezení D bloku
        {
            Generace náhodné x pozice D bloku
            Generace náhodné y pozice D bloku

            Pro různé typy transformací D bloku
            {
                Podvzorkování D bloku na velikost R bloku
                Srovnání R bloku s D blokem na generované pozici
            }
        }
    }
}
```

### 6.1.1 Geometrické transformace doménového bloku

Pro lepší výsledky hledání co nejpodobnějších doménových bloků se na doménové bloky, které jsou zmenšené na velikost oblastních bloků, aplikují různé geometrické transformace. Tyto transformace jsou složeny převážně z překlápění a rotací. Zpravidla se aplikuje osm transformací při každém porovnání oblastního bloku s doménovým:

1. identita
2. otočení kolem středu bloku o 90° po směru hodinových ručiček
3. otočení kolem středu bloku o 90° proti směru hodinových ručiček
4. otočení kolem středu bloku o 180° proti směru hodinových ručiček
5. otočení kolem střední vertikální osy bloku
6. otočení kolem střední horizontální osy bloku

7. otočení kolem první diagonály bloku
8. otočení kolem druhé diagonály bloku

Tyto geometrické transformace spolu s podvzorkováním doménového bloku (tedy jeho zmenšením na velikost bloku oblastního) plní úlohu koeficientů afinní transformace $a_i, b_i, c_i, d_i$ ze vztahu (4.3) .

### 6.1.2 Transformační struktura

Jsou-li provedeny veškeré geometrické transformace, je možné podle vztahů (4.9) a (4.10) vypočítat škálovací faktor $s$ a posun $o$. Rozdílnost oblastního a doménového bloku určuje střední kvadratická odchylka (RMSE = Root Mean Square Error), která je definována jako:

$$RMSE = \sqrt{E} \tag{6.1}$$

Kde $E$ je kvadratická Euklidovská vzdálenost ze vztahu (4.7). Nejvíce vhodný doménový blok je určen nejmenší hodnotou střední kvadratické odchylky. Proto vždy, když je nalezena menší hodnota $RMSE$, zaznamenají se veškeré potřebné informace o tomto doménovém bloku.

Tyto potřebné informace se zaznamenávají do strukturovaného datového modelu, který se dá považovat za afinní transformaci $w_i$ ze vztahu (4.3). Tato struktura totiž obsahuje vše, co je potřeba pro popis jednoho oblastního bloku. Nazvěme ji *transformační strukturou*.

Použitá transformační struktura má následující tvar:

```
18 struct Transform
19 {
20     _int8 domainPosX;
21     _int8 domainPosY;
22     float scale;
23     float offset;
24     _int8 transformType;
25 }
```

První dvě 8-bitové integer proměnné *domainPosX* a *domainPosY* slouží k uložení koordinátů doménového bloku, 32-bitové proměnné *scale* a *offset* s plovoucí řádovou čárkou slouží k uložení škálovacího faktoru a posunu a do 8-bitové integer proměnné *transformType* se ukládá číslo od 0 do 7 – to reprezentuje použitou geometrickou transformaci, která byla shledána nejlepší ze všech transformací uvedených v kapitole 6.1.1.

Pro popis všech oblastních bloků a tedy i celé pracovní matice, se struktury ukládají do pole. Počet transformačních struktur v tomto poli je tedy stejný jako počet oblastních bloků v signálu. Je tedy patrné, že míra komprese závisí na počtu těchto transformačních struktur, resp. počtu oblastních bloků. Tento počet lze regulovat nastavováním velikostí oblastních bloků. Zpravidla je velikost oblastních bloků 2x2, 4x4, 8x8 nebo 16x16. Tyto rozměry bývají většinou vypočítány na základě nastavené velikosti doménového bloku a kontraktivního faktoru.

Po nalezení nejvhodnějšího doménového bloku ke každému oblastnímu bloku, a tedy po naplnění všech transformačních struktur, je kódování dokončeno.

## 6.2 Dekódování

Se získanými transformačními strukturami, které popisují mapování doménových bloků do bloků oblastních, můžeme celý signál dekódovat. Na počátku tedy máme nulovou pracovní matici o daném rozměru. Na každý oblastní blok této matice aplikujeme příslušné operace. Parametry potřebné pro provedení těchto operací jsou obsaženy v transformační struktuře.

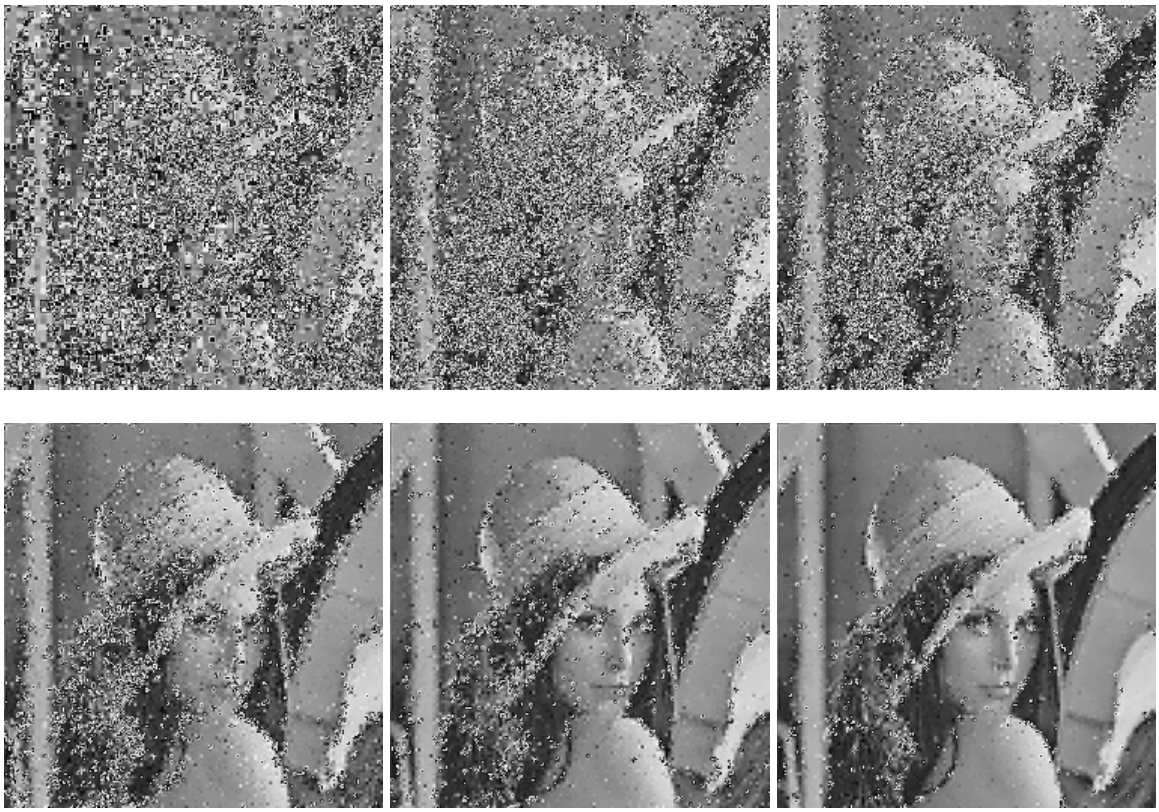
Symbolicky lze celý postup dekódování zapsat následovně:

```
1 Pro všechny R bloky ve sloupci
2 {
3     Pro všechny R bloky v řádku
4     {
5         Načtení D bloku na souřadnicích domainPosX a domainPosY
6         Geometrická transformace D bloku podle transformType
7         Podvzorkování D bloku na velikost R bloku
```

```
 8          Každá hodnota D bloku vynásobena scale a přičten offset
 9      }
10  }
```

Dekódování spočívá v iterativní aplikaci tohoto postupu, protože každou iterací se zvyšuje rozlišení pracovní matice. Mapovaný doménový blok se totiž stává součástí jiného doménového bloku, který se při jiné transformaci opět mapuje a tím se zvyšuje detail.

Na Obr. 24. je zobrazeno prvních 9 iterací dekódování obrazu „Lena“. Pro ukázku je po každé iteraci aplikována zpětná waveletová transformace, aby bylo vidět, jak je každou iterací obraz rekonstruován.



Obr. 24. Iterativní dekódování obrazu „Lena“

## 6.3 Uložení do souboru

Aby mohl být tento kompresní algoritmus použitelný v praxi, musí existovat prostředek, jak takto zakódovaná data uchovat pro archivaci nebo případný přenos po komunikačním kanálu. Data je tedy nutné uložit do souboru s určitou strukturou. Kromě uložení samotných transformační struktur je také pro potřeby dekódování znát základní údaje o původním obrazu a některých parametrech, které byly pro kódování použity. Tyto informace lze všechny sloučit do jedné datové struktury nazývané *FileHeader*, tedy *hlavička souboru*.

Její tvar je následující:

```
11 struct FileHeader
12 {
13     char waveletBasis[5];
14     _int16 imageWidth;
15     _int16 imageHeight;
16     _int8  domainWidth;
17     _int8  domainHeight;
18 }
```

Do pole znaků *waveletBasis* se ukládá identifikátor bázové funkce, která byla před kódováním použita pro dekompozici obrazu a tudíž má být po dekódování použita pro jeho rekonstrukci. Další dvě 16-bitové integer proměnné *imageWidth* a *imageHeight* poskytují informaci o šířce a výšce komprimovaného obrazu. 8-bitové integer proměnné *domainWidth* a *domainHeight* udávají velikost (tedy šířku a výšku) doménového bloku, která byla použita při kódování.

Po uložení hlavičky souboru následuje ukládání všech transformačních struktur. Tyto struktury se ukládají za sebou jako posloupnost binárních dat.

Při případném otevření souboru je tedy nejprve načtena hlavička souboru a získány základní informace o dekódovaném obrazu. Díky těmto informacím lze před samotným dekódovacím algoritmem připravit všechny potřebné pracovní matice a proměnné. Teprve poté můžou být spuštěny příslušné algoritmy zajišťující rekonstrukci dat z transformačních struktur.

# 7 VÝSLEDKY EXPERIMENTŮ

Popisovaný algoritmus byl experimentálně vyzkoušen na klasickém 256x256px obrázku „Lena“ v šedé škále. Byla sledována kvalita výsledného obrazu při různých kompresních poměrech. K waveletové transformaci a dyadickému rozkladu první úrovně byl použit mateřský wavelet Daubechies 2.řádu, který je z hlediska výpočetní náročnosti a dosažených výsledků optimální. Následující série obrázků ukazuje srovnání fraktálního algoritmu s využitím waveletové transformace (FIC) s nejvíce používaným algoritmem pro ztrátovou kompresi – JPEG. Výsledná kvalita obrazu je měřena PSNR ukazatelem, jehož hodnoty jsou pro různé kompresní poměry (CR) uvedeny v Tab.1.

Obr. 25. Originální obrázek „Lena“

Obr. 26. Aplikace FIC s CR = 4.36:1

Obr. 27. Aplikace JPEG s CR = 4.36:1

Obr. 28. Aplikace FIC s CR=17.45:1

Obr. 29. Aplikace JPEG s CR=17.42:1

Obr. 30. Aplikace FIC s CR=34.91:1

Obr. 31. Aplikace JPEG s CR=34.73:1

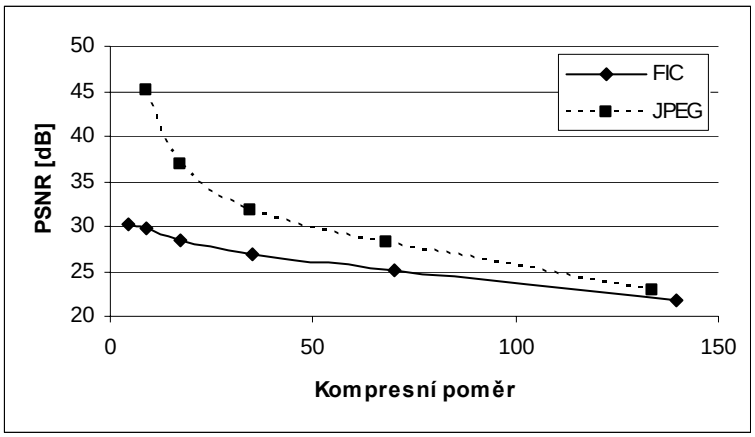
Obr. 32. Aplikace FIC s CR=69.83:1

Obr. 33. Aplikace JPEG s CR=68.09:1

Tab. 1. Srovnání FIC a JPEG u obrazu „Lena“

| | | | | | |
|---|---|---|---|---|---|
| **FIC** | **Kompresní poměr CR** | 4.36:1 | 17.45:1 | 34.91:1 | 69.83:1 |
| | **PSNR [dB]** | 30.13 | 28.54 | 26.79 | 25.03 |
| **JPEG** | **Kompresní poměr CR** | 4.36:1 | 17.42:1 | 34.73:1 | 68.09:1 |
| | **PSNR [dB]** | - | 45.12 | 31.79 | 28.30 |

Na Obr.34. je zobrazena grafická závislost kvality obrazu na několika různých kompresních poměrech. U obrázku „Lena“ lze vidět, že JPEG si i přes vyšší kompresní poměry zachovává slušnou kvalitu obrazu. Při použití FIC již určité detaily zmizely a začalo se objevovat velké zkreslení některých částí obrazu. V tomto případě tedy FIC nepodává tak dobré výsledky, jak se mohlo očekávat.




Obr. 34. Závislost PSNR na kompresním poměru u obrazu „Lena“ u FIC a JPEG

Srovnání kompresních technik JPEG a FIC bylo provedeno i na jiných typech obrázků. Pro ukázku byl vybrán obrázek obsahující přírodní útvary, které jsou fraktálům velmi příbuzné a dá se tudíž předpokládat, že výsledky v případě FIC budou velmi příznivé. Následující série obrázků ukazuje komprimaci obrazu s přírodní scenérií při vyšších kompresních poměrech.

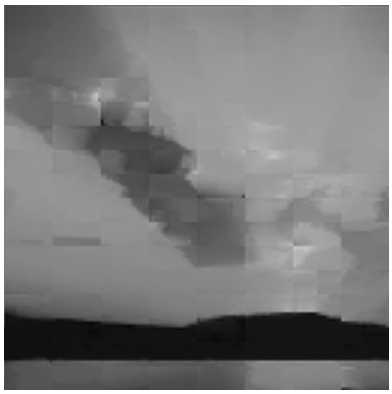
Obr. 35. Originální obrázek „Západ slunce“

Obr. 36. Aplikace FIC s CR = 34.91:1

Obr. 37. Aplikace JPEG s CR = 34.69:1

Obr. 38. Aplikace FIC s CR = 69.83:1

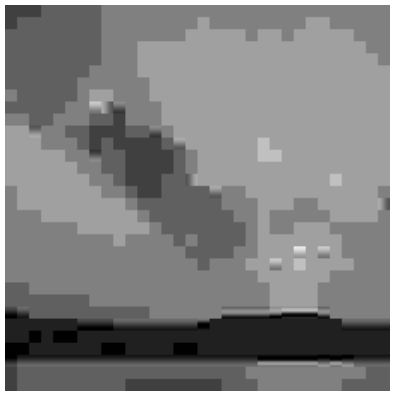
Obr. 39. Aplikace JPEG s CR = 69.49:1



Obr. 40. Aplikace FIC s CR=139.67:1

Obr. 41. Aplikace JPEG s CR=140.97:1

Obr. 42. Aplikace FIC s CR=279.34:1



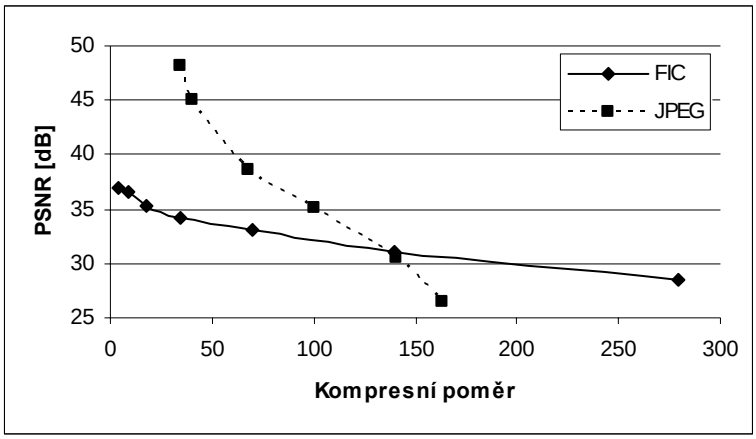
Obr. 43. Aplikace JPEG s CR=163.34:1

Tab. 2. Srovnání FIC a JPEG u obrazu „Západ slunce“

| | | | | | |
|---|---|---|---|---|---|
| **FIC** | **Kompresní poměr CR** | 34.91:1 | 69.83 | 139.67:1 | 279.34:1 |
| | **PSNR [dB]** | 34.15 | 33.07 | 31.14 | 28.54 |
| **JPEG** | **Kompresní poměr CR** | 34.69:1 | 69.49 | 140.97:1 | 163.34:1 |
| | **PSNR [dB]** | 48.13 | 36.36 | 30.57 | 26.54 |

Na Obr.44. je zobrazena grafická závislost kvality obrazu na několika různých kompresních poměrech. U obrázku “Západ slunce“ lze vidět, že při kompresním poměru 140:1 (bod průniku obou křivek) a vyšším podává metoda FIC mnohem lepší kvalitu obrazu než JPEG. Také zkreslení obrazu po komprimaci je u FIC přirozenější než u JPEG.

Algoritmus FIC je tedy vhodný pro zpracování statických obrazů, které obsahují přírodní útvary a lze ho tedy používat např. u digitálních fotografií exteriérů.




Obr. 44. Závislost PSNR na kompresním poměru u obrazu „Západ slunce“ u FIC a JPEG

Pro ukázku, v čem může být waveletová transformace nápomocná, byl srovnán jednoduchý klasický fraktální algoritmus (CFC) s fraktálním algoritmem, který využívá waveletovou transformaci (FIC). Protože se kompresní poměry u fraktálních algoritmů pohybují skokově, nebylo možno statické obrazy zkomprimovat na stejném kompresním poměru. Přesto byla snaha přiblížit kompresní poměry obou algoritmů co nejblíže k sobě, aby byla vyjádřena rozdílná kvalita. Pro experimenty byl opět použit obraz „Západ slunce“.

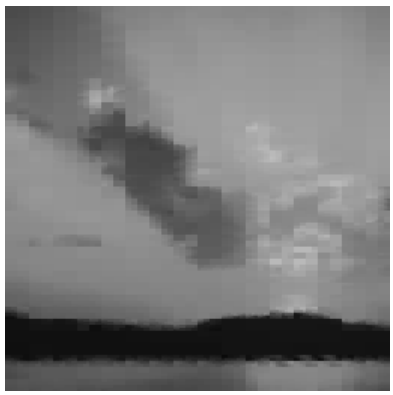
Obr. 45. Aplikace FIC s CR=8.72:1

Obr. 46. Aplikace CFC s CR=9.60:1

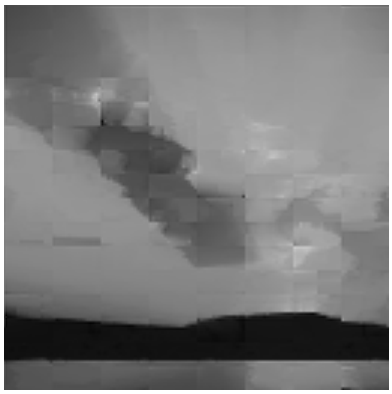
Obr. 47. Aplikace FIC s CR=34.91:1

Obr. 48. Aplikace CFC s CR=38.41:1

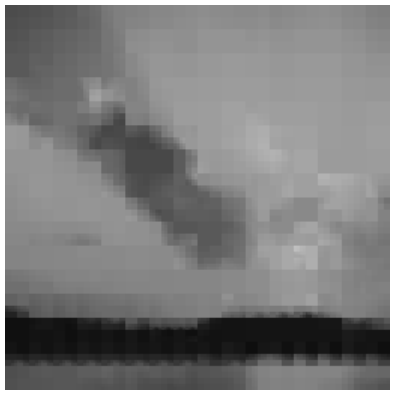
Obr. 49. Aplikace FIC s CR=69.83:1



Obr. 50. Aplikace CFC s CR=76.82:1



Obr. 51. Aplikace FIC s CR=139.67:1

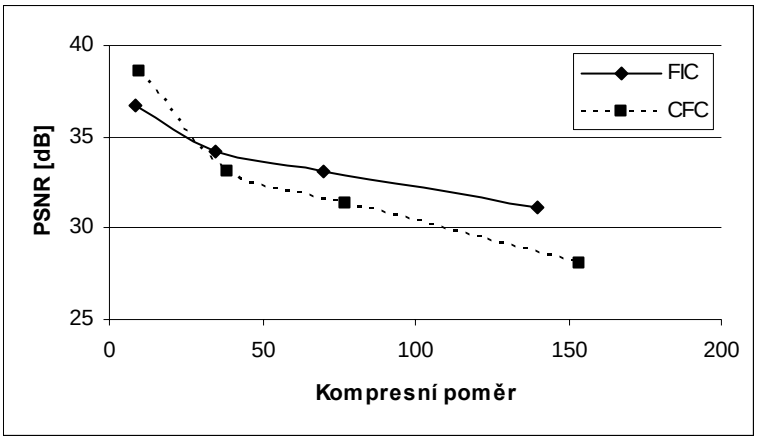


Obr. 52. Aplikace CFC s CR=153.64:1

Tab. 3. Srovnání FIC a CFC u obrazu „Západ slunce“

| | | | | | |
|---|---|---|---|---|---|
| **FIC** | **Kompresní poměr CR** | 8.72:1 | 34.91:1 | 69.83 | 139.67:1 |
| | **PSNR [dB]** | 36.66 | 34.15 | 33.07 | 31.14 |
| **CFC** | **Kompresní poměr CR** | 9.60:1 | 38.41:1 | 76.82 | 153.64:1 |
| | **PSNR [dB]** | 38.58 | 33.07 | 31.31 | 28.08 |

Na Obr.53. lze vidět, že FIC dosáhlo u vyšších kompresních poměrů mnohem lepších výsledků než klasická fraktální komprese CFC. To je dáno především tím, že waveletová transformace s použitím báze Daubechies 2.řádu zajišťuje hladkou rekonstrukci obrazu a tím také mizí některé blokové artefakty, které vznikly mapováním doménových bloků do bloků oblastních. Je nutné také zmínit, že kódování u CFC je několikanásobně výpočetně náročnější a trvá tedy mnohem delší dobu, protože se prohledává celá oblast obrazu. V případě FIC se prohledává pouze LL pásmo dyadicky rozloženého obrazu a proto je počet prováděných operací mnohem nižší. Pokud se tedy vezmou v úvahu tato fakta, lze říci, že FIC je pro zpracování obrazů efektivnější než CFC.




Obr. 53. Závislost PSNR na kompresním poměru u obrazu „Západ slunce“ u FIC a CFC

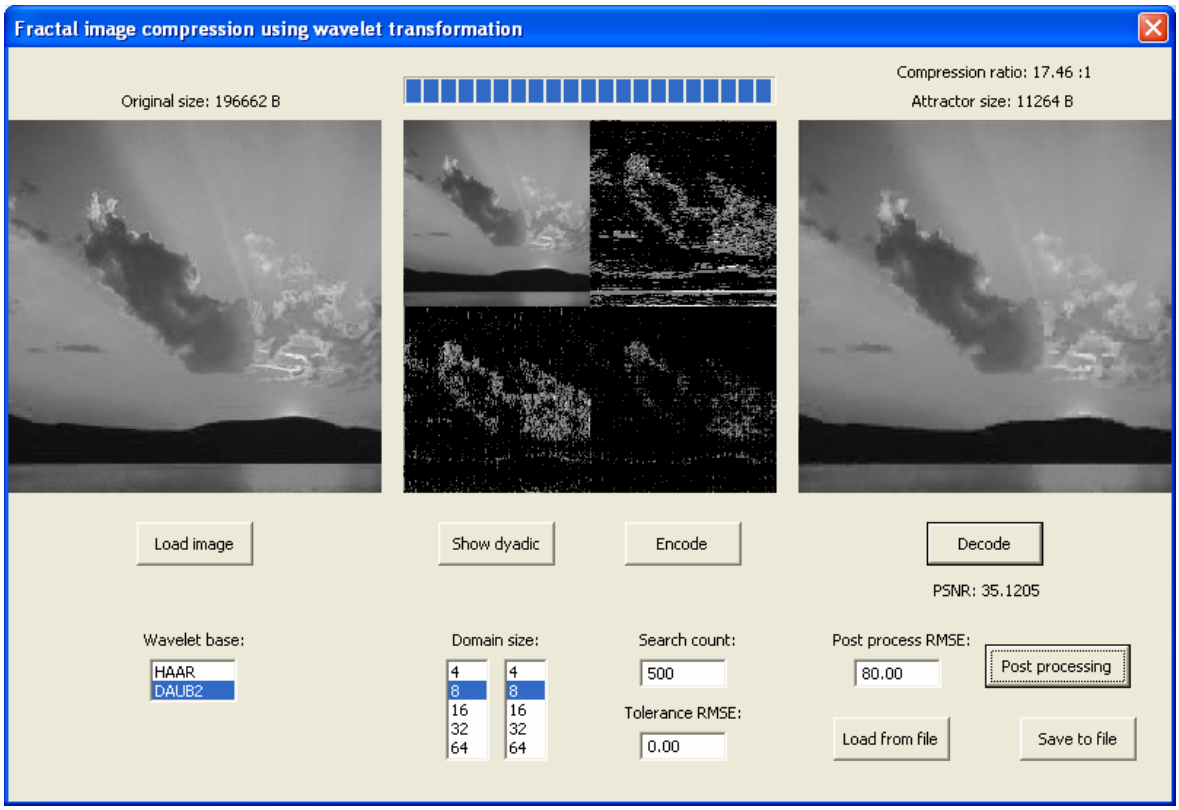
Bylo provedeno mnoho experimentů s několika statickými obrazy. Většina prokázala, že FIC se hodí především pro zpracování exteriérových obrazů na vyšších poměrech. Pro

subjektivní hodnocení lidským zrakem jsou v příloze PI ukázány některé další zpracované statické obrazy na různých kompresních poměrech.

# 8 DEMONSTRACE PROGRAMU

Celý program pro fraktální kompresi obrazu s využitím waveletové transformace byl napsán v jazyce C++ s podporou open-sourcové multiplatformní API knihovny wxWidgets. Tato knihovna využívá pro vytváření grafického uživatelského rozhranní (GUI) nativní ovládací prvky dané platformy. Program byl zkompilován pro platformu Windows 95/98/ME/NT/2000/XP.

GUI programu s rozmístěnými ovládacími prvky je ukázáno na Obr.54.




Obr. 54. GUI programu

Lze vidět, že program zobrazuje 3 bitmapy a několik ovládacích prvků:

- tlačítko *Load image* – kliknutím se zobrazí dialogové okno se seznamem souborů, které lze otevřít. Tyto soubory mohou být ve formátu BMP. Po výběru souboru se dialogové okno uzavře a první bitmapa vlevo se vyplní obrazovými daty uloženými v příslušném souboru.

- select-box *Wavelet base* – slouží k výběru bázové funkce, která se má použít pro dyadický rozklad a rekonstrukci obrazu.
- tlačítko *Show dyadic* – po kliknutí se zobrazí ve středové bitmapě dyadický rozklad obrazu první úrovně.
- select-boxy *Domain size* – zde si lze zvolit šířku a výšku doménového bloku. Toto nastavení ovlivňuje výsledný kompresní poměr, protože velikost oblastního bloku je dána kontrakcí doménového bloku. Čím menší budou doménové bloky, tím menší budou oblastní bloky a tím větší počet oblastních bloků (resp. transformačních struktur) bude zapotřebí, aby pokryly celý signál.
- edit-box *Search count* – zadává se počet pokusů o nalezení nejvhodnějšího doménového bloku, tedy počet generací pozic doménových bloků k jednomu oblastnímu. Tento parametr ovlivňuje výpočetní náročnost algoritmu a také výslednou kvalitu komprimovaného obrazu.
- edit-box *Tolerance RMSE* – zadává se číslo RMSE, které bude u doménového bloku tolerováno. Pokud střední kvadratická odchylka doménového bloku od bloku oblastního bude menší než tolerovaná RMSE, pak se tento doménový blok může použít a další doménové bloky se již hledat nebudou (nebudou tedy generovány všechny pozice zadané parametrem *Search count*). Tento parametr ovlivňuje výpočetní náročnost algoritmu a také výslednou kvalitu komprimovaného obrazu.
- tlačítko *Encode* – po kliknutí se spustí fraktální kódovací algoritmy. Průběh kódování bude zaznamenáván v progress-baru nad středovou bitmapou dyadicky rozloženého obrazu. Po ukončení kódovaní se také vypočítá velikost kompresního poměru a zobrazí se nad třetí bitmapou vpravo.
- tlačítko *Decode* – kliknutím se provede jedna iterace dekódování a ve třetí bitmapě vpravo bude prováděna rekonstrukce obrazu. Při každé iteraci se aktualizuje PSNR ukazatel kvality.
- edit-box *Post Process RMSE* – zadává se číslo RMSE, se kterým se bude pracovat u post processingových operací.
- tlačítko *Post processing* – po kliknutí se spustí vyhlazování rekonstruovaného obrazu. Vypočítává se střední kvadratická odchylka mezi všemi pixely originálního

a rekonstruovaného obrazu. Pokud je tato odchylka větší než číslo z edit-boxu *Post Process RMSE*, hodnota pixelu v rekonstruovaném obrazu se nahradí zprůměrňovanou hodnotou jeho sousedních pixelů.

- tlačítko *Save to file* – kliknutím se zobrazí dialogové okno pro uložení základních údajů a samotných transformačních struktur do souboru. Při ukládání se tento soubor označí extenzí *.FIC*.
- tlačítko *Load from file* – kliknutím se zobrazí dialogové okno se soubory, které mají extenzi *.FIC*. Po výběru souboru se z něj načte jeho hlavička se základními údaji a poté se do paměti načtou i samotné transformační struktury. Bude také spuštěna první iterace dekódování.

# ZÁVĚR

Tato diplomová práce popisuje kompresní algoritmus, který je založen na klasických fraktálních PIFS algoritmech, ovšem pro kvalitnější zpracování signálu se používá waveletová transformace, která má za úkol celý signál analyzovat. Úkolem komprese je pak nalézt sobě-podobnost waveletových koeficientů, které se nacházejí v nízko-frekvenčním pásmu dyadicky rozloženého statického obrazu.

Zde popsané algoritmy byly implementovány v jazyce C++ a s pomocí knihovny wxWidgets, která se stará o vytvoření grafického uživatelského rozhranní. Výsledný program umožňuje uživateli zadávat různé parametry ovlivňující míru komprese a kvalitu výsledného obrazu. Transformační struktury, které se získají fraktálním kódováním a které popisují LL pásmo dyadicky rozloženého signálu, lze ukládat do (resp. načítat ze) souboru s definovaným datovým formátem. Tímto způsobem může být takto zkomprimovaný statický obraz přenášen po komunikačním kanálu.

Provedené experimenty ukázaly, že je fraktální komprese vhodná zejména pro obrazy exteriérů, ve kterých jsou obsaženy přírodní útvary podobné fraktálům. Pokud jsou tyto obrazy zpracovány na vysokých kompresních poměrech, dosahuje se lepších výsledků než v případě JPEG komprese. Také zkreslení, které je vysokým kompresním poměrem vyvoláno, působí z hlediska subjektivního hodnocení mnohem přirozeněji. V opačných případech bývá JPEG flexibilnější a to je také jeden z důvodů, proč se přednostně využívá.

Z uživatelského pohledu na kompresní metody nemá fraktální komprese příliš velké využití také pro svou silnou asymetričnost. Její přínos tedy spočívá především u složitějších analýz a zpracování signálů, kde díky jejím vlastnostem je možné provádět takové operace, jako přibližovat nebo oddalovat určitou část obrazu (zoom) nebo také zvětšovat či zmenšovat celé obrazy, aniž by se objevovaly nežádoucí jevy pixelizace. Využitím waveletové transformace je možné nejen urychlit kódovací algoritmy až několikanásobně, protože lze prohledávat jen ty části obrazu, které jsou opravdu zapotřebí, ale lze také docílit ještě mnohem kvalitnějších výstupů než v případě klasické fraktální komprese.

# ZÁVĚR V ANGLIČTINĚ

This diploma work describes compression algorithm based on classic fractal PIFS algorithms, with use of the wavelet transformation for better signal processing. While the aim of using wavelet transformation is to analyze the signal, the aim of the fractal compression is to find self-similarity of wavelet coefficients, which are found in low-frequency sub-band of dyadic decomposed static image.

The algorithms described above were implemented in C++ language with the wxWidgets library that creates graphical user interface. The program created allows user to set different parameters influencing the compression ratio and quality of the final image. The transformation structures which are gained by fractal encoding and are describing LL sub-band, could be saved to (loaded from) file with defined data format. The compressed static image can be transferred through communicating channel by this way.

The experiments made proved, that fractal compression is suitable for the outside images, which contain natural formation similar to fractals. High ratio processed images this way gain better results then with JPEG compression. The distortion that is produced by high compression ratio is also much more natural, when subjectively evaluated. In other cases JPEG is usually more flexible that is one of the reasons why JPEG is used more often.

Using the fractal compression has no reason when considering the user's point of view, especially for its strong asymmetry. Its main contribution lies in complicated analysis and signal processing, where it is possible to do operation such as zoom-in or zoom-out particular part of the image, extend or reduce the whole image without undesirable effect of pixelization. Using the wavelet transformation is possible not only to speed up encoding algorithms several times, because of searching through the parts of image that are really necessary, but also make more superior outputs, than with use of classic fractal compression.

# SEZNAM POUŽITÉ LITERATURY

[1] Šmíd, R. (2001) *Úvod do vlnkové transformace* [online]. Dostupný na WWW: http://measure.feld.cvut.cz/usr/staff/smid/wavelets/wavelet-intro-html.html

[2] Klecker, D. (2007) *Optimalizace kritérii waveletové transformace pro kompresi a filtraci signálu*. Disertační práce, VŠB – Technická univerzita Ostrava.

[3] Valens, C. (2001) *A Really Friendly Guide to Wavelets* [online]. Dostupný na WWW: http://perso.orange.fr/polyvalens/clemens/wavelets/wavelets.html

[4] Kiselev, A. (2004) *Fundamentals of the Wavelet Transform Theory* [online]. Dostupný na WWW: http://www.basegroup.ru/filtration/intro-to-wavelets.en.htm

[5] Tišnovský, P. (2005) *Fraktály v počítačové grafice I,II* [online]. Dostupný na WWW: http://www.root.cz/clanky/fraktaly-v-pocitacove-grafice-i/

[6] Kominek, J. (2001) *Introduction to Fractal compression* [online]. Dostupný na WWW: http://www.faqs.org/faqs/compression-faq/part2/section-8.html

[7] Sheng, Y. (2000). *Wavelet transform.* CRC Press LLC.

# SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

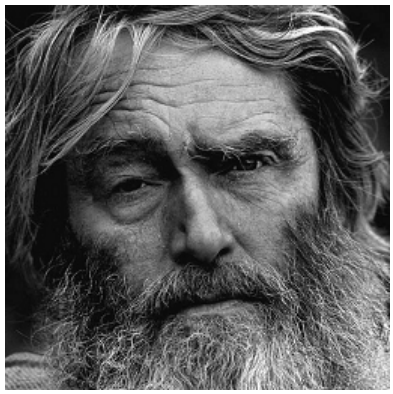
| | |
|---|---|
| STFT | Short-Time Fourier Transformation |
| FFT | Fast Fourier Transformation |
| WT | Wavelet Transformation |
| CWT | Continuous Wavelet Transformation |
| DWT | Discrete Wavelet Transformation |
| FDWT | Fast Discrete Wavelet Transform |
| FIR | Finite Impulse Response |
| BW | Band Width |
| HP | Horní Propust |
| DP | Dolní Propust |
| IFS | Iterated Function Systems |
| PIFS | Partitioned Iterated Function Systems |
| LL | Low-Low |
| LH | Low-High |
| HL | High-Low |
| HH | High-High |
| FIC | Fractal Image Compression |
| JPEG | Joint Photographic Experts Group |
| PSNR | Peak Signal to Noise Ratio |
| CR | Compression Ratio |
| CFC | Classical Fractal Compression |
| GUI | Graphical User Interface |
| RMSE | Root Mean Square Error |

# SEZNAM OBRÁZKŮ

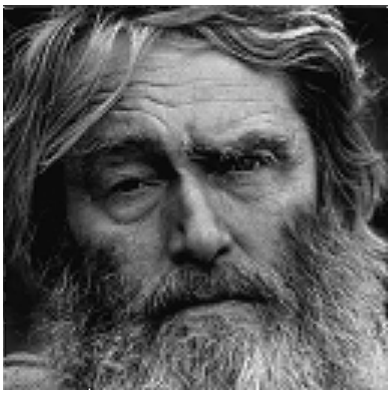
# SEZNAM TABULEK

# SEZNAM PŘÍLOH

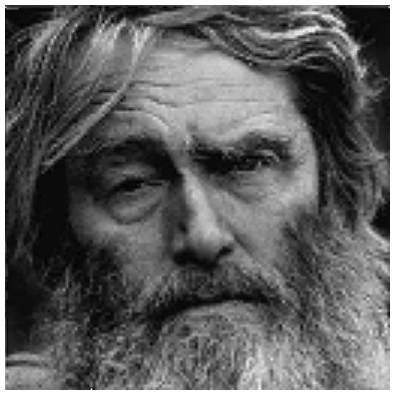
# PŘÍLOHA P I: DALŠÍ FIC EXPERIMENTY



Originální obrázek



Kompresní poměr 4.36:1



Kompresní poměr 8.73:1

Kompresní poměr 17.46:1

Kompresní poměr 34.92:1

Kompresní poměr 69.84

Kompresní poměr 139.67:1

Originální obrázek

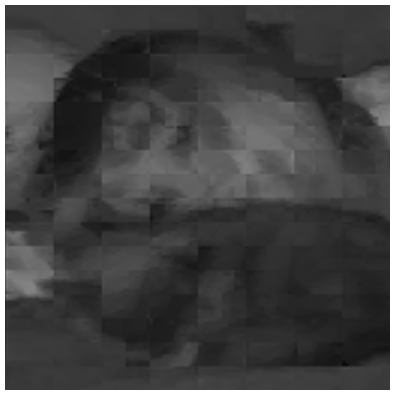
Kompresní poměr 4.36:1

Kompresní poměr 8.73:1

Kompresní poměr 17.46:1

Kompresní poměr 34.92:1

Kompresní poměr 69.84



Kompresní poměr 139.67:1

Originální obrázek

Kompresní poměr 4.36:1

Kompresní poměr 8.73:1

Kompresní poměr 17.46:1

Kompresní poměr 34.92:1

Kompresní poměr 69.84

Kompresní poměr 139.67:1

Originální obrázek

Kompresní poměr 4.36:1

Kompresní poměr 8.73:1

Kompresní poměr 17.46:1

Kompresní poměr 34.92:1

Kompresní poměr 69.84

Kompresní poměr 139.67:1