Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Disertační práce

2007 **Ing. Karel Uhlíř**

Západočeská univerzita v Plzni
Fakulta aplikovaných věd

Aplikace radiálních bázových funkcí v počítačové grafice a zpracování obrazu

Ing. Karel Uhlíř

disertační práce
k získání akademického titulu doktor
v oboru Informatika a výpočetní technika

Školitel: Prof. Ing. Václav Skala, CSc.

Katedra: Informatiky a výpočetní techniky

Plzeň 2007

University of West Bohemia
Faculty of Applied Sciences

Application of radial basis functions in computer graphics and image processing

Ing. Karel Uhlíř

Dissertation submitted
for the Degree of Doctor of Philosophy
in Computer Sciences

Supervisor: Professor Václav Skala

Department of Computer Sciences and Engineering

Czech Republic, Pilsen 2007

Prohlášení

Prohlašuji, že jsem disertační práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne

_____

podpis

Poděkování

Chtěl bych poděkovat všem za podporu při práci na této disertační práci. V první řadě rodině, přátelům, kolegům a svému vedoucímu Prof. Ing. Václavu Skalovi, CSc.

Abstract

This work is about extension of knowledge about Radial basis functions method and its application. This method is very strong and useful for visualization of point cloud data and during last few years its more and more important because of very good results in cases that are problematic for other methods.

Main goal of this work is in usage of RBF method in image processing. Reconstruction of corrupted images is part of image processing which is very important in many scientific areas and the RBF method should be successfully used in this area. For example reconstruction of partial images received from satellites, retouching of inpainted images of removing of company logo from video sequence. In all these cases should be used the RBF method and results are very good. The RBF method is able to solve problems that are otherwise solvable with several independent methods.

You can fined several new methods in this work that are using the RBF method for reconstruction of corrupted images. All methods are compared and with other standards methods too. Important part is selection of base function. Stress is put on reconstruction with minimal error and on simple usage of the method because with selection of the basis function number of variable parameters which has influence on quality of reconstruction is increasing.

Abstrakt

Tato práce je zaměřena na prohloubení znalostí Radial Basis Function (RBF) metody, která je velmi silným nástrojem pro vizualizace objektů definovaných shlukem bodů. Této metodě je v poslední době věnována velká pozornost, neboť jako interpolační/aproximační metoda dokáže produkovat velmi zajímavé výsledky i v případech, kdy ostatní metody selhávají.

Hlavním cílem práce je využití možností RBF metody v oblasti zpracování obrazu, kde jsou problémy, na které by mohla být metoda s úspěchem použita. Konkrétně oblast zabývající se rekonstrukcí poškozených obrazů je dnes velmi důležitá v mnoha vědních oborech. Jako příklad můžeme uvést rekonstrukce částečných obrazů obdržených z družic během vzdáleného pozorování vesmírných těles, retušování inpaintingu z fotografií nebo odstraňování loga společnosti při přebírání části vysílání jiného televizního kanálu. Ve všech těchto problémech lze s úspěchem použít RBF metodu, která většinou poskytuje výborné výsledky a zvládá i problémy, na které by bylo nutné použít několik rozdílných metod.

V této práci naleznete několik nově navržených metod využívající RBF metodu k rekonstrukci poškozených obrazů. Všechny nové metody jsou porovnány jak vůči sobě, tak s jinými standardními metodami. Důležitou částí je volba bázové funkce vhodné k rekonstrukci. Důraz je kladen na rekonstrukci s minimální chybou a na jednoduchost použití metody neboť s volbou bázové funkce přibývají volitelné parametry ovlivňující kvalitu rekonstrukce.

Obsah

1 ÚVOD 1

1.1 INTERPOLACE DAT 1
1.2 METODY INTERPOLACE 1

2 RADILA BASIS FUNCTION (RBF) METODA 3

2.1 ÚVOD 3
2.2 ZÁKLADNÍ RBF METODA 5
2.3 ROZŠÍŘENÁ RBF METODA 7
2.4 ŘEŠENÍ A ŘEŠITELNOST LINEÁRNÍHO SYTÉMU 9
2.5 BÁZOVÉ FUNKCE 10
2.6 POUŽITÍ RBF METODY 14
2.7 URYCHLOVACÍ TECHNIKY 16
2.7.1 Fast Multipole Method 16
2.7.2 Redukce bodů 17
2.7.3 Výběr podmnožiny bodů 17
2.7.4 Volba metody na řešení systému rovnic 18
2.8 ALGORITMICKÁ SLOŽITOST 18
2.8.1 Vytvoření systému lineárních rovnic 18
2.8.2 Řešení systému lineárních rovnic 19
2.8.3 Vyhodnocení interpolační funkce 19
2.9 VÝHODY A NEVÝHODY METODY 19
2.9.1 Výhody 19
2.9.2 Nevýhody 20
2.10 PŘÍKLADY 20
2.10.1 Vědecké vizualizace 20
2.10.2 Vizualizace lékařských dat 21
2.10.3 Vizualizace naskenovaných 3D dat 21
2.10.4 Příklad vizualizace 21

3 REKONSTRUKCE POŠKOZENÝCH OBRAZŮ 23

3.1 ÚVOD 23
3.2 POUŽITÍ RBF METODY VE ZPRACOVÁNÍ OBRAZU 24
3.3 DEFINICE PROBLÉMU 24
3.4 TESTOVANÉ OBRAZY A MASKY 26
3.5 METODY POROVNÁNÍ REKONSTRUKCE OBRAZU 26
3.6 REKONSTRUKCE OBRAZU RBF METODOU 28
3.6.1 RBF metoda - globální přístup 30
3.6.2 RBF metoda - lokální přístup 32
3.6.3 Analýza rekonstrukce 33
3.7 NAVRŽENÉ METODY PRO REKONSTRUKCI POŠKOZENÝCH OBRAZŮ 38
3.7.1 Scan-line metoda – přímá rekonstrukce 39
3.7.2 Scan-line metoda – rekonstrukce zleva, zprava 41
3.7.3 Scan-line metoda – vícestranná 43
3.7.4 Scan-line metoda – oboustranná 48
3.7.5 Maximální informace v k-okolí 49
3.8 DETEKCE POŠKOZENÝCH OBRAZOVÝCH BODŮ 52

3.9 POROVNÁNÍ METOD 53
3.9.1 Výsledky rekonstrukce jednotlivých metod 54
3.9.2 Porovnání metod rekonstrukce 63
3.10 POROVNÁNÍ S EXISTUJÍCÍMI METODAMI REKONSTRUKCE POŠKOZENÝCH OBRAZŮ 64
3.10.1 Inpainting, škrábance, šum 64
3.10.2 CSRBF 67
4 ZÁVĚR 68
LITERATURA 68
PŘÍLOHA A – BÁZOVÉ FUNKCE 74
PŘÍLOHA B – INTERPOLACE RŮZNÝMI BÁZOVÝMI FUNKCEMI 76
PŘÍLOHA C – REKONSTRUKCE POŠKOZENÍ 88
PŘÍLOHA D – SEZNAM POUŽITÝCH BÁZOVÝCH FUNKCÍ 93
PŘÍLOHA E – SEZNAM TESTOVANÝCH OBRAZŮ 94
PŘÍLOHA F – METODY 99
PŘÍLOHA G – UŽIVATELSKÁ DOKUMENTCE 100
PŘÍLOHA H – PUBLIKACE, POBYTY A KONFERENCE 103

Tabulky

Tabulka 1. Základní typy bázových funkcí. 11
Tabulka 2. Compactly-Supported radiální funkce [Wendland95]. 13
Tabulka 3: Označení chybových obrazů. 28
Tabulka 4: Statistika rekonstrukce globální metodou. 31
Tabulka 5: Testování metody přímé rekonstrukce. 40
Tabulka 6: Testování metody rekonstrukce zleva. 42
Tabulka 7: Změna strany rekonstrukce. 43
Tabulka 8: Vyhodnocení rekonstrukce pro Obrázek 46. 44
Tabulka 9: Testování metody rekonstrukce zleva-zprava. 45
Tabulka 10: Testování metody rekonstrukce zleva-shora. 46
Tabulka 11: Testování metody rekonstrukce shora-zdola. 46
Tabulka 12: Testování metody rekonstrukce zleva-shora-zprava-zdola. 47
Tabulka 13: Testování metody rekonstrukce zleva-zprava-shora-zdola. 48
Tabulka 14: Testování metody oboustranné rekonstrukce. 49
Tabulka 15: Testování metody maximální délky vektoru. 52
Tabulka 16: Označení metod rekonstrukce. 53
Tabulka 17: Porovnání metod rekonstrukce na obrazu č.1. 54
Tabulka 18: Porovnání metod rekonstrukce na obrazu č.2. 55
Tabulka 19: Porovnání metod rekonstrukce na obrazu č.3. 55
Tabulka 20: Porovnání metod rekonstrukce na obrazu č.4. 56
Tabulka 21: Porovnání metod rekonstrukce na obrazu č.5. 57
Tabulka 22: Porovnání metod rekonstrukce na obrazu č.6. 57
Tabulka 23: Porovnání metod rekonstrukce na obrazu č.7. 58
Tabulka 24: Porovnání metod rekonstrukce na obrazu č.8. 58
Tabulka 25: Porovnání metod rekonstrukce na obrazu č.9. 59
Tabulka 26: Porovnání metod rekonstrukce na obrazu č.10. 59
Tabulka 27: Porovnání metod rekonstrukce na obrazu č.11. 60
Tabulka 28: Porovnání metod rekonstrukce na obrazu č.12. 60
Tabulka 29: Porovnání metod rekonstrukce na obrazu č.13. 61
Tabulka 30: Porovnání metod rekonstrukce na obrazu č.14. 61
Tabulka 31: Porovnání metod rekonstrukce na obrazu č.15. 62
Tabulka 32: Vyhodnocení nejlepší metody rekonstrukce pro daný obraz. 63
Tabulka 33: Porovnání rekonstrukce obrazu 75a. 65
Tabulka 34: Porovnání rekonstrukce pomocí RBF s rekonstrukcí Bertalmiho metodou na obrazu 76a. 66
Tabulka 35: Časová náročnost rekonstrukce. 66

Algoritmy

Algoritmus 1: Zjednodušený návrh algoritmu rekonstrukce obrazu. 30
Algoritmus 2: Globální rekonstrukce poškozeného obrazu. 31
Algoritmus 3: Lokální rekonstrukce poškozeného obrazu. 33
Algoritmus 4: Scan-line metoda přímé rekonstrukce. 39
Algoritmus 5: Scan-line metoda rekonstrukce zleva. 42
Algoritmus 6: Scan-line metoda rekonstrukce z různých stran obrazu (zleva-zprava). 44
Algoritmus 7: Scan-line metoda rekonstrukce koncového a počátečního obrazového bodu delší díry. 48
Algoritmus 8: Maximální informace z k-okolí. 51

Obrázky

Obrázek 1: Interpolace dat různými po částech hladkými bázovými funkcemi. (vlevo – interpolace všech hodnot, vpravo – výřez a označení bázových funkcí).....11
Obrázek 2: Interpolace dat různými hladkými bázovými funkcemi. (vlevo – interpolace všech hodnot, vpravo – výřez).....12
Obrázek 3: Průběh CSRBF funkcí uvedených v Tabulka 2. 13
Obrázek 4: Rozdílné hodnoty parametru α pro CSRBF.....14
Obrázek 5: Ukázka generování dodatečných bodů ve směru normály.....15
Obrázek 6: Rekonstrukce ruky definované množinou bodů se správnou (b) a špatnou (c) definicí dodatečných bodu. [Carr01] 15
Obrázek 7: Průřez prsty ruky rekonstruované ze shluku bodů na Obrázek 6. [Carr01].....16
Obrázek 8: Ukázka redukce počáteční množiny bodů. [Carr01].....17
Obrázek 9: Popsání shluku bodů (438,000 bodů) radiální bázovou funkcí. [Carr01].....19
Obrázek 10: Vizualizace geofyzikálních dat. [ARANZ].....20
Obrázek 11: Definice povrchu určeného několika body. 21
Obrázek 12: Vizualizace povrchu podle definice. 21
Obrázek 13: Vizualizace povrchu *marching triangles* metodou z rovnice (2.10) a TPS bázovou funkcí.....22
Obrázek 14: Různé druhy poškození obrazu a) text, b) šum, c) škrábance. 23
Obrázek 15: Poškozený obraz inpaintingem (vlevo) a obraz po rekonstrukci Bertalmiho metodou (vpravo).....24
Obrázek 16: Poškozený obraz (vlevo) a opravený obraz (vpravo) [Savchenko02]. 24
Obrázek 17: Definice oblasti Ω ve vztahu k hodnotám matice A ((2.12), (2.14)). 25
Obrázek 18: Poškozený černobílý (vlevo) a barevný obraz (vpravo). 26
Obrázek 19: Vizualizace MSE dle (3.7). Chyba se odliší jasem pixelu. (zleva: S, S^2 , 255 - S, 255 - S^2).....27
Obrázek 20: Porovnání vizualizace MSE chyby (vlevo a uprostřed) a chybového obrazu (vpravo) dle (3.10). 28
Obrázek 21: Originální obraz - a) jako povrch b) jako obraz. 29
Obrázek 22: Poškozený obraz - a) jako povrch b) jako obraz. 29
Obrázek 23: Různé přístupy k zpracování obrazu. 30
Obrázek 24: Vyplnění submatice A při použití TPS bázové funkce (vlevo) a *compactly-supported* bázové funkce (vpravo). Nenulové hodnoty jsou tmavé a nulové hodnoty světlé. 30
Obrázek 25: Obrázek rekonstruovaný globální metodou – a) originální obraz, b) poškozený obraz, c) rekonstruovaný obraz, d) S^2 obraz.....31
Obrázek 26: Znázornění chyby rekonstrukce. 32
Obrázek 27: Definice okolí poškozené části, které je nutné dodržet pro korektní rekonstrukci u *compactly-supported* bázových funkcí. 32
Obrázek 28: Definice části obrazu s jasovými hodnotami (vlevo) a jejich interpolace RBF metodou (vpravo).....34
Obrázek 29: Řez interpolovanými jasovými hodnotami z Obrázek 29. 34
Obrázek 30: Interpolace dat GRBF s nevhodným parametrem ε ($\varepsilon = 5$). 35
Obrázek 31: Interpolace dat MQ bázovou funkci s různými hodnotami parametru ε. (vlevo $\varepsilon = 0.1$) 35
Obrázek 32: Interpolace dat *quantic* bázovou funkci. 36
Obrázek 33: Interpolace CSRBF s nevhodně nastaveným poloměrem parametrem α. 36
Obrázek 34: Různé interpolace dat TPS bázovou funkcí: černá – všechny hodnoty, červená – bez zakroužkované hodnoty se znalostí okolí, modra – bez

zakroužkované hodnoty pouze v řezu. 37
Obrázek 35: Klasické čtvercové okolí zpracovávaného bodu. 37
Obrázek 36: Různá alternativní okolí zpracovávaného bodu. 38
Obrázek 37: Volba způsobu rekonstrukce poškozeného obrazového bodu. a) námi zvolený způsob ve středu oblasti interpolace – 16 známých hodnot, b) možný způsob se znalostí většího počtu hodnot – 24 známých hodnot. 38
Obrázek 38: Základní konfigurace okna na začátku rekonstrukce. 38
Obrázek 39: Kroky metody přímé rekonstrukce: a) rekonstrukce prvního poškozeného bodu – 23 známých hodnot, b) zde je již použita hodnota z předcházejícího kroku – 23 známých hodnot, c) další postup metody – 22 známých hodnot. 39
Obrázek 40: Ukázka rekonstrukce přímou metodou obrazu 1. (zleva: originální obraz, poškozený obraz, rekonstruovaný obraz, B2 obraz) 40
Obrázek 41: Nejlépe zrekonstruovaný obraz z Tabulky 4 přímou metodou. (zleva: originální obraz, poškozený obraz, rekonstruovaný obraz, B2 obraz) 40
Obrázek 42: Kroky metody postupné rekonstrukce zleva: a) pokud je za sebou více poškozených ob., tak algoritmus pokračuje na dalším řádku, b) další větší poškozená část obrazu, c) pokračování rekonstrukce v dalším průchodu. 41
Obrázek 43: Ukázka postupné rekonstrukce obrazu zleva/zprava. (Obraz č.5) 42
Obrázek 44: Kroky metody postupné rekonstrukce zleva a zprava: a) pokud je za sebou více poškozených ob., tak algoritmus zleva pokračuje na dalším řádku, b) algoritmus zprava však nemá v této oblasti problémy, c) a d) problematické místo pro algoritmus zleva i zprava, e) v dalším průchodu je problematické místo již zcela opraveno. 43
Obrázek 45: Rekonstrukce poškozeného obrazu (vlevo) metodou zprava-zleva (uprostřed) a metodou shora-zdola (vpravo). 44
Obrázek 46: Ukázka postupné rekonstrukce obrazu zleva-zprava. 45
Obrázek 47: Obraz s největší chybou rekonstrukce. (zleva: originální obraz, poškozený obraz, rekonstruovaný obraz, B2 obraz) 45
Obrázek 48: Ilustrativní obrázek metody zleva-shora. (Obraz č.6) 46
Obrázek 49: Ilustrativní obrázek průběhu metody shora-zdola. (Obraz č.5) 47
Obrázek 50: Ilustrativní obrázek průběhu LTRB metody. (Obraz č.15) 47
Obrázek 51: Ilustrativní obrázek průběhu oboustranné metody. (Obraz č.1) 49
Obrázek 52: Histogramy počtu výskytů délky vektoru známých hodnot v 24-okolí (vlevo: vstupní obraz, vpravo: 2620 iterace rekonstrukce) 50
Obrázek 53: Rekonstrukce obrazu k němuž je uveden histogram. (vlevo: vstupní obraz, uprostřed: 1310 iterace rekonstrukce, vpravo: 2620 iterace rekonstrukce, Obraz č.1) 50
Obrázek 54: Histogram počtu výskytu shluků určitých délek při rekonstrukci. (Obraz č.1) 50
Obrázek 55: Rekonstrukce obrazu č.15 metodou maximální informace v k-okolí. 52
Obrázek 56: Označení poškozené části obrazu a) automaticky například vyplněním oblasti se stejnou intenzitou obrazových bodů b) ručně vytvořená maska. 53
Obrázek 57: Rekonstrukce obrazu č.1 s 60% poškozením TPS bázovou funkcí a TB metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 54
Obrázek 58: Rekonstrukce obrazu č.2 s 19% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 54
Obrázek 59: Rekonstrukce obrazu č.3 s 25% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz

W2) 55
Obrázek 60: Rekonstrukce obrazu č.4 s 47% poškozením TPS bázovou funkcí a OI metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 56
Obrázek 61: Rekonstrukce L, LR, LT, MAX a globální metodou. 56
Obrázek 62: Rekonstrukce obrazu č.5 s 60% poškozením TPS bázovou funkcí a LT metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 56
Obrázek 63: Rekonstrukce obrazu č.6 s 20% poškozením TPS bázovou funkcí a TB metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 57
Obrázek 64: Rekonstrukce obrazu č.7 s 27% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 57
Obrázek 65: Rekonstrukce obrazu č.8 s 60% poškozením TPS bázovou funkcí a LR metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 58
Obrázek 66: Rekonstrukce obrazu č.9 s 20% poškozením TPS bázovou funkcí a TB metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 58
Obrázek 67: Rekonstrukce obrazu č.10 s 28% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 59
Obrázek 68: Rekonstrukce obrazu č.11 s 60% poškozením TPS bázovou funkcí a LR metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 59
Obrázek 69: Rekonstrukce obrazu č.12 s 20% poškozením TPS bázovou funkcí a TB metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 60
Obrázek 70: Rekonstrukce obrazu č.13 s 28% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 61
Obrázek 71: Rekonstrukce obrazu č.14 s 6.5% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2) 61
Obrázek 72: Rekonstrukce obrazu č.15 s 60% poškozením TPS bázovou funkcí a LR metodou. (vlevo - poškozený obraz, vpravo - rekonstruovaný obraz) 62
Obrázek 73: Rekonstrukce obrazu č.15 s 60% poškozením TPS bázovou funkcí a LR metodou. (rozdílový obraz W2) 62
Obrázek 74: Bertalmiho testovací obrazy 64
Obrázek 75: Bertlamiho rekonstrukce obrazu s kočárem (vlevo) a naše rekonstrukce (vpravo) nejlepší metodou MAX. 64
Obrázek 76: Bertalmiho rekonstrukce skokana na laně (vlevo), naše rekonstrukce metodou MAX (uprostřed) a detaily na rekonstruovanou část obrazu (vpravo – originál, Bertalmio, MAX). 65
Obrázek 77: Bertalmiho rekonstrukce rytířů (vlevo) a naše rekonstrukce (vpravo). 65
Obrázek 78: Filtrace poškozeného obrazu: a) poškozený obraz, b) maximální propust, ... 66
Obrázek 79: Rekonstrukce obrazu s vážkou. (a) poškozený obraz, b) rekonstrukce CSRBF, c) rekonstrukce TPS metodou MAX) 67

Značení a zkratky

$\underline{x}$	vrchol, (x, y, z) v E^3
x, y, z	souřadnice vrcholu
ϕ	radiální bázová funkce
$\phi(r)$	hodnota radiální bázové funkce
a_{ij}	prvek matice
$\mathbf{A}$	čtvercová matice
λ_j	váha bázové funkce
r	hodnota euklidovské vzdálenosti dvou bodů
$\boldsymbol{\lambda}$	vektor vah
$\mathbf{h}$	vektor pravých stran
C^k	spojitost
γ_k	váha k-tého stupně polynomu
p_k	polynom stupně k
$\mathbf{P}$	matice polynomů
$\boldsymbol{\gamma}$	vektor vah polynomů
$\mathbf{0}$	nulový vektor
$\mathbf{B}$	matice lineárního systému
ε	konstanta
α	poloměrem působnosti bázové funkce
d_i	hodnota off-surface bodu
$\| \ \|$	euklidovská norma (vzdálenost)

RBF	Radial Basis Functions
CSRBF	Compactly Supported Radial Basis Function
SLAE	System of Linear Algebraic Equations
TPS	Thin-plate Spline
PSNR	Peak Signal to Noise Ratio
MSE	Mean Square Error
FEM	Final Elements Method
CT	Computer Tomography
MRI	Magnetic Resonation
FMM	Fast Multipole Method
SVD	Singular Value Decomposition
GMRES	Generalized Minimal Residual Method

1 ÚVOD

1.1 Interpolace dat

Společným problémem v mnoha vědních oborech je interpolace roztroušených (neuspořádaných) dat, tj. problém proložení hladké křivky respektive hladkého povrchu roztroušenými nebo nerovnoměrně rozloženými vzorky. Metody pro interpolaci vzorků dat jsou potřebné v mnoha vědních oborech, kde jsou měřeny nebo generovány body na různých nebo náhodných pozicích. Cílem interpolace je zjištění funkce, která svým průběhem bude vystihovat rozložení všech bodů. Získanou funkci bude možné vyhodnotit v libovolném bodě a získat tak potřebné chybějící hodnoty pro zpracování problému. Interpolace na rozdíl od aproximace danými body prochází.

Jako základní zdroje roztroušených dat můžeme označit hodnoty naměřené fyzikální veličiny, experimentální výsledky a hodnoty získané výpočtem. Takto získané hodnoty můžeme nalézt v mnoha různých vědních oborech. Tak například nerovnoměrně naměřené hodnoty fyzikálních veličin jsou získávány v geologii, meteorologii, oceánografii, kartografii nebo těžbě. Roztroušená experimentální data jsou produkována v chemii, fyzice a strojírenství a nerovnoměrně roztroušené vypočtené hodnoty vznikají například jako výstup metody konečných prvků (FEM) a aplikací počítačové grafiky, vizualizace dat nebo zpracování obrazu.

Tyto vědní obory využívají interpolaci roztroušených bodů pro získání hodnot v libovolné pozici a ne pouze v té, ve které jsou data známa. Odhad dalších hodnot pomáhá při velmi častém zobrazování vícerozměrných roztroušených dat. Například v medicíně je interpolace roztroušených dat základem pro rekonstrukci povrchu orgánů z CT nebo MRI obrazů, kdy řezy získané uvedenými metodami jsou použity k dopočítání chybějící informace mezi nimi.

Přes velké množství metod a způsobů interpolace roztroušených dat zůstává je tento proces velmi obtížný a výpočetně náročný. Literatura se v této oblasti zabývá různými aspekty interpolace například výpočetní složitostí, hladkostí, přípustnou distribucí dat nebo časovou náročností.

1.2 Metody interpolace

Existuje celá řada publikací, které se zabývají interpolací roztroušených vzorků dat. Jedna z prvních prací, která se zabývá tímto problémem je založené na inverzní vzdálenostní váze dat. Označuje se jako Shepardova metoda [Shepard68]. Shepard definoval C^0 spojitou interpolační funkci jako vážený průměr dat s váhami inverzně úměrnými vzdálenosti. Metoda je globální a v případě přidání, odebrání nebo modifikování libovolného vzorku dat, musí být všechny váhy přepočítány. Franke a Nelson představili modifikovanou kvadratickou Shepardovu metodu [Franke82] řešící uvedené problémy a vytvářející C^1 spojitou interpolaci.

Další řešení interpolačního problému je přes metody konečných prvků. Tento přístup je založen na vytvoření určitého typu optimální triangulace na množině datových bodů k vymezení lokálního okolí, přes které jsou záplaty definovány. Záplaty jsou omezeny tak, aby interpolovali originální data. Základní pravidla pro optimální triangulaci navrhl Lawson [Lawson77] . Hlavním pravidlem je, že tenké trojúhelníky s malými úhly jsou nepřípustné. Po částech lineární aproximace přes triangulaci není hladká, ale má pouze C^0 spojitost. Nejpoužívanější C^1 spojitá metoda používá Clough-Tocherův trojúhelníkový

interpolant [Clough65, Barnhill77, Goshtasby87]. Některé metody, které jsou citlivé na rozložení dat, nemohou podmínku tenkého trojúhelníku s malými úhly dodržet.
V posledních několika letech zaznamenala velký rozmach interpolační metoda založená na definici interpolační funkce jako lineární kombinace radiálně symetrických bázových funkcí. Každá z funkcí je soustředěná na datovou hodnotu. Neznámé hodnoty bázové funkce jsou získány vyřešením soustavy lineárních rovnic.

V oblasti zpracování obrazu je interpolace roztroušených dat používána k rekonstrukci mezi uniformními i neuniformními vzorky. Množství různých metod je uvedeno například v [Glassner95]. Trend v současných algoritmech je v použití hierarchického nebo vícerozměrné filtrování k rozšíření známé informace z roztroušených vzorků do neznámých oblastí. Burt navrhl hierarchické polynomiální filtrování k získání množiny vícerozměrných low-pas filtrovaných obrazů, které mohou být kombinovány do hladkého povrchu, který prochází originálními daty [Burt88]. Mitchelli navrhl vícestupňové filtrování k manipulaci s vzorky s velmi proměnnou hustotou [Mitchelli86] a v Lee [Lee97] je použita hierarchie vrstev generujících B-spline funkce, které jsou následně spojeny váhovou funkcí do jedné. Bertalmio [Bertalmio00] představil zajímavou metodu na rekonstrukci poškozených obrazů, která je založena na parciálních diferenciálních rovnicích. I v oblasti zpracování obrazu se v posledních letech začalo využívat metody založené na lineární kombinaci symetrických bázových funkcí.

2 RADILA BASIS FUNCTION (RBF) METODA

2.1 Úvod

Základ interpolační metodě založené na lineární kombinaci posunující se jednoduché bázové funkce, které je radiálně symetrická okolo svého středu dal L.R.Hardy [Hardy71] a nazval ji *radial basis function* (RBF) metoda. Hardy vytvořil *multquadric* metodu, aby řešil problém z kartografie. Snažil se o vytvoření spojité funkce, která by procházela naměřenými hodnotami v terénu a mohl tak vytvořit automatický postup pro generování vrstevnicových map. Standardním postupem v té době bylo, že zkušený topograf vzal naměřené hodnoty a snažil se odhadnou, jak asi daný terén vypadá. Mohlo se tedy stát, že dva různí topografové vytvořili dvě rozdílné mapy ze stejného zdroje naměřených dat. První pokusy o vytvoření topografické plochy směřovaly k použití Fourierovy a polynomiální metody. Avšak obě tyto metody se ukázali jako nevhodné. Interpolace Fourierovou řadou nebyla akceptovatelná neboť výstupní množina dat měla velké sklony ke kmitání mezi vzorovými vzorky dat. Polynomiální interpolace zase nebyla schopna postihnout rychlé změny terénu. Další metodou, která byla vyzkoušena, byla metoda nejmenších čtverců, která však exaktně nevystihovala naměřená data. Protože Hardy nenašel mezi existujícími metodami vhodnou metodu na vyřešení jeho problému, tak začal hledat nové řešení. Nejdříve začal analyzovat jedno dimenzionální problém a nalezl, že tvar může být reprezentován po částech lineární interpolační funkcí. Pro množinu n různých zdrojových bodů $\{x_j\}_{j=1}^{n}$ a korespondujících měření $\{f_j\}_{j=1}^{n}$ navrhl následující formu interpolační funkce

$$s(x) = \sum_{j=1}^{n} \lambda_j \cdot |x - x_j|, \tag{2.1}$$

kde λ_j jsou určeny uspořádáním, například $s(x_j) = f_j, j = 1,2,...,n$. Geometricky to znamená interpolaci dat lineární kombinací n posunutí absolutní hodnoty bázové funkce $|x|$, kde vrchol každé bázové funkce je umístěn do jednoho ze vstupních bodů.

Hardymu nevyhovovala skokově se měnící průběh rovnice (2.1) a nahradil ji tedy absolutní hodnotu bázové funkce spojitě diferencovatelnou funkcí. Použil funkci $\sqrt{c^2 + x^2}$, kde c je libovolné nenulová konstanta. Rovnice (2.1) se tedy změnila na

$$s(x) = \sum_{j=1}^{n} \lambda_j \cdot \sqrt{c^2 + (x - x_j)^2}\ . \tag{2.2}$$

Pro $c = 0$ je rovnice (2.2) ekvivalentní rovnici (2.1). Na takto popsaná data mohly být bez problému použity i metody pro zjišťování maxima a minima funkce, což se dalo uplatnit v topologii při hledání vrcholů a údolí. Další výhodou metody bylo možné použití ve více než jedné dimenzi. Absolutní hodnota rozdílu mezi dvěma jedno dimenzionálními body je jednoduše Euklidovská vzdálenost mezi dvěma body ($|x - x_j| = \sqrt{(x - x_j)^2}$). Máme-li tedy n nezávislých zdrojových bodů $\{(x_j, y_j)\}_{j=1}^{n}$ a korespondující topologické měření $\{f_j\}_{j=1}^{n}$, navrhl Hardy proložení dat funkcí

$$s(x,y)=\sum_{j=1}^{n}\lambda_j\cdot\sqrt{(x-x_j)^2+(y-y_j)^2}\,, \tag{2.3}$$

kde λ_j jsou opět určeny rozkladem a $s(x_j,y_j)=f_j, j=1,2,...,n$. Geometricky tato funkce koresponduje s interpolací dat lineární kombinací *n* posunů kužele. To znamená radiálně symetrickou funkcí $\phi(r)=r$, kde $r=\sqrt{x^2+y^2}$. Vrchol každého kužele je centrován na jeden ze zdrojových bodů.

Stejně jako jedno dimenzionální případ, tak i uvedený dvou dimenzionální trpí tím, že jeho výsledkem je po částech spojitá funkce. Stejně jako v jedno dimenzionálním případě, tak i zde navrhl Hardy úpravu, která tento problém vyřešila a nová forma interpolantu dostala tvar

$$s(x,y)=\sum_{j=1}^{n}\lambda_j\cdot\sqrt{c^2+(x-x_j)^2+(y-y_j)^2}\,. \tag{2.4}$$

Pro $c\neq 0$ je tato funkce nekonečně diferencovatelná. Tato technika nazývaná *multivariable calculus* může být použita pro určení vlastností topologického povrchu. Funkce je aproximační.

Hardy zjistil, že funkce (2.3) je výborná pro aproximaci povrchu z roztroušených, řídkých dat. Pojmenoval tuto novou techniku *multiquadric method* (MQ). Hardy tuto metodu původně vyvinul pro řešení dvou-dimenzionálního interpolačního problému, ale také si uvědomil, že stejně jako pro dvě dimenze, může být metoda rozšířena do libovolné dimenze. Následuje tedy definice MQ metody pro interpolaci multi-dimenzionálních roztroušených dat:

Definice 1 (MQ metoda) Je dána množina *n* nezávislých bodů (nebo zdrojových bodů) $\{\underline{x}_j\}_{j=1}^{n}$ v R^d ($\underline{x}_j=(x_1^{(j)},x_2^{(j)},...,x_d^{(j)})$), a korespondující (skalární) hodnoty $\{f_j\}_{j=1}^{n}$, tak MQ interpolant dat je dán jako

$$s(\underline{x})=\sum_{j=1}^{n}\lambda_j\cdot\sqrt{c^2+\left\|\underline{x}-\underline{x}_j\right\|^2}\,, \tag{2.5}$$

kde $\|\cdot\|$ je Euklidovská norma. Koeficienty λ_j jsou určeny z interpolační podmínky $s(\underline{x}_j)=f_j, j=1,...,n$, což vede k následujícímu lineárnímu systému:

$$\begin{bmatrix} & \mathbf{A} & \end{bmatrix}\begin{bmatrix}\boldsymbol{\lambda}\end{bmatrix}=\begin{bmatrix}\mathbf{f}\end{bmatrix}, \tag{2.6}$$

kde matice **A** je tvořena $a_{j,k}=\sqrt{c^2+\left\|\underline{x}_j-\underline{x}_k\right\|^2}$.

Po publikaci Hardyho MQ metody v [Hardy71] začalo mnoho výzkumníků používat tuto metodu v mnoha vědních oborech. Metoda byla například použita v hydrologii, geodesii, fotometrii a geologii. Více informací můžete nelézt v [Hardy90].

Vedle použití metody pro nové aplikace bylo uvažováno mnoho dalších vlastností metody. Současně bylo nezávisle vyvinuto mnoho nových metod podobných MQ metodě. Tyto nové metody měly stejnou základní formu jako (2.5) , ale namísto použití stejné bázové funkce používali jinou rozdílnou funkci, která byla však stále závislá na Euklidovské vzdálenosti. Například Duchon [Duchon77] použil bázovou funkci $r^2 \log r$ a r^3, kde $r = \|\underline{x}\|$. Jeho funkce $r^2 \log r$ se označuje jako *thin-plate spline* (TPS). *Thin-plate splines* jsou odvozeny z minimalizace integrálu křivosti mezi interpolačními funkcemi přes oblast roztroušených vzorků dat. Jsou hojně používány pro svoje vizuální výsledky a stabilitu pro velká data. Ačkoli jsou obvykle formulovány jako řešení variačního problému, Duchon ukázal, že *thin-plate splines* mohou být odvozeny z radiálních bázových funkcí [Duchon77].

Schagen [Schagen79] použil Gaussovu funkci $e^{-(\varepsilon r)^2}$, kde ε je volný parametr a Hardy a Gopfer [Hardy75] použili jako bázovou funkci $(c^2 + r^2)^{-1/2}$.

Jednu z nejdůležitějších studií MQ metody udělal Franke [Franke82, Franke79], který otestoval velké množství metod na interpolaci roztroušených dat na vzorky 6 různých funkcí, které byly nevzorkovány s různou hustotou. Testoval zhruba 30 různých metod a zjistil, že MQ metoda je z nich nejzajímavější a poskytuje nejlepší výsledky. V 13 z 18 testů vyšla nejlépe a ve 3 testech ze zbylých 5 skončila na druhém místě.

Přesto, že Franke prováděl mnoho experimentálních testování metody, tak jeho metoda neměla matematické pozadí jako metody Duchone [Duchon77] nebo Schagena [Schagen79]. Nebylo například známo, zda je systém jednoznačně řešitelný, například pokud byl lineární systém (2.6) nesingulární. Frankeho numerické experimenty vedli k domněnce, že MQ metoda byla vždy nesingulární, ale důkaz nebyl nikdy proveden.

Matematické podklady dal MQ metodě až Micchelli [Micchelli86]. Mimo to, že provedl matematický důkaz Frankeho domněnek, tak i doplnil metodu o podmínky, které garantovali nesingularitu pro různé bázové funkce jako například Hardyho a Gopfera.

Skoro 15 let po představení Hardyho myšlenky se spojením multi-dimenzionálních roztroušených dat s lineární kombinací posunující se bázové funkce (2.5) , ukázal Micchelli, že metoda je bezpodmínečně nesingulární, ale také garantoval nesingularitu pro mnoho jiných bázových funkcí. Bylo zjištěno, že Hardyho MQ metoda byla jen jedním specifickým příkladem mnohem obecnější metody. Hlavní myšlenkou obecné metody je použití jedné posunující se bázové funkce $\phi(r)$, která je závislá pouze na Euklidovské vzdálenosti od svého středu. Funkce s touto jednoduchou závislostí je samozřejmě kruhově (*radially*) symetrická okolo svého středu, proto je tato bázová funkce označována jako „*radial function*“ nebo „*radial basis function*“ (RBF) a metoda se tedy nazývá RBF metoda.

2.2 Základní RBF metoda

RBF metoda je jednou ze základních metod pro interpolaci vícerozměrných roztroušených dat. Můžeme se setkat s dvěmi základními formami metody. Postupně si představíme obě formy a uvedeme si i vlastnosti jednotlivých metod.

První „základní“ RBF metoda je definována takto:

Definice 2. (základní RBF metoda) Je dána množina n od sebe různých bodů $\{\underline{x}_j\}_{j=1}^{n}$ korespondujících s hodnotami $\{h_j\}_{j=1}^{n}$ v těchto bodech. Základní interpolační předpis je potom dán jako

$$f(\underline{x}) = \sum_{j=1}^{n} \lambda_j \cdot \phi\left(\left\|\underline{x} - \underline{x}_j\right\|\right), \tag{2.7}$$

kde $\phi\left(\left\|\underline{x} - \underline{x}_j\right\|\right)$ budeme zapisovat jako $\phi(r), r \geq 0$ a ϕ je radiální funkce. Koeficienty λ_j jsou určeny z interpolačních podmínek $f\left(\underline{x}_j\right) = h_j, j = 1,...,n$, které vedou k následujícímu lineárnímu systému:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_n \end{bmatrix} \text{ tedy} \tag{2.8}$$

$$\begin{bmatrix} \\ \mathbf{A} \\ \\ \end{bmatrix} \begin{bmatrix} \\ \boldsymbol{\lambda} \\ \\ \end{bmatrix} = \begin{bmatrix} \\ \mathbf{h} \\ \\ \end{bmatrix}, \tag{2.9}$$

kde $a_{j,k} = \phi\left(\left\|\underline{x}_j - \underline{x}_k\right\|\right), k = 1,...,n$.

Jak již bylo řečeno, Micchelli [Micchelli86] dal dostatečné podmínky pro $\phi(r)$ v rovnici (2.7) k zjištění toho, že matice **A** v rovnici (2.9) je bezpodmínečně regulární a základní RBF metoda má jednoznačné řešení.

První dostačující podmínky pro $\phi(r)$ zajišťující regulárnost matice **A** a tedy její jednoznačné řešení dal již v roce 1938 Schoenberg [Schoenberg38]. Mitechelli pouze později ukázal, že tyto podmínky mohou být rozšířeny na větší množinu bázových funkcí. Pro pochopení výsledků Schoenberga a Micchelliho je potřeba nadefinovat ryze monotónní funkci.

Definice 3. [Wright03] (ryze monotónní funkce) Řekneme, že funkce ψ je ryze monotónní na intervalu $[0,\infty)$ pokud platí, že

1. $\psi \in C[0,\infty)$
2. $\psi \in C^{\infty}[0,\infty)$
3. $(-1)^l \psi^{(l)}(r) \geq 0$ pro $r > 0$ a $l = 0,1,2,\ldots$

Věta 1. (Schoenberg [Schoenberg38]) Jestliže $\psi(r) = \phi(\sqrt{r})$ je ryze monotónní avšak ne konstantní na $[0,\infty)$. Potom pro libovolnou množinu n rozdílných bodů $\{\underline{x}_j\}_{j=1}^{n}$ je matice **A** o rozměru $n \times n$ pozitivně definitní a tedy regulární.

Z uvedené věty můžeme vyvodit, že základní RBF metoda je jednoznačně řešitelná pro bázové funkce jako například Gaussovu $\phi(r) = e^{-(\varepsilon r)^2}$, inverzní kvadratickou $\phi(r) = \frac{1}{1+(\varepsilon r)^2}$ nebo inverzní multiquadric $\phi(r) = \frac{1}{\sqrt{1+(\varepsilon r)^2}}$. Pro multiquadric bázovou funkci to neplatí. Micchelli tedy uvedl další větu, která rozšířila původní větu 1.

Věta 2. (Micchelli [Micchelli86]) Nechť $\psi(r) = \phi(\sqrt{r}) \in C^0[0,\infty), \psi(r) > 0$ pro r > 0 a $\psi'(r)$ je ryze monotónní na $(0,\infty)$. Potom pro libovolnou množinu *n* rozdílných bodů $\{\underline{x}_j\}_{j=1}^n$ je matice **A** o rozměru $n \times n$ regulární. Mimo to, pro $n \geq 2$ má matice $n-1$ negativních vlastních čísel a jedno pozitivní.

Uvedená Věta 2. už popsala i MQ bázovou funkci a zajistila jednoznačnou řešitelnost systému matice **A**. Nečekaně však Věta 2. neplatila pro velmi oblíbenou „thin-plate spline" (TPS) bázovou funkci. Pro zahrnutí TPS do Věty 2. musela být tato věta upravena a museli být přidány další dodatečné podmínky.

Pokud totiž řekneme, že máme množinu bodů takových, že *n* > 1, $\{\underline{x}_j\}_{j=1}^n$ v R^{n-1} a každý bod je vrcholem jednotkového simplexu, tak všechny položky matice **A** jsou nulové a matice je tedy singulární.

2.3 Rozšířená RBF metoda

Ještě před definicí rozšířené RBF metody je nutné provést ještě dodatečnou definici.

Definice 4. ($\coprod_m(R^d)$) Nechť $\coprod_m(R^d)$ je prostor všech polynomů d-proměnných které mají stupeň menší nebo roven *m*. Mimoto nechť *M* udává dimenzi $\coprod_m(R^d)$. Potom $M = \binom{m+d}{m}$.

Definice 5. [Wright03, Micchelli86](rozšířená RBF metoda) Je dána množina *n* od sebe různých bodů $\{\underline{x}_j\}_{j=1}^n$ korespondujících s hodnotami $\{h_j\}_{j=1}^n$ v těchto bodech. Rozšířený interpolační předpis je potom dán jako

$$f(\underline{x}) = \sum_{j=1}^{n} \lambda_j \cdot \phi\left(\left\|\underline{x} - \underline{x}_j\right\|\right) + \sum_{k=1}^{M} \gamma_k p_k(\underline{x}), \ \mathbf{x} \in R^d \qquad \textbf{(2.10)}$$

kde $\{p_k(\underline{x})\}_{k=1}^M$ je báze pro $\coprod_m(R^d)$ a $\phi(r), r \geq 0$ je libovolná radiální funkce. Aby bylo možné použít polynomiální výraz, tak musely být doplněny následující omezujíc podmínky:

$$\sum_{j=1}^{n} \lambda_j p_k(\underline{x}_j) = 0, k = 1,2,...,M \ . \qquad \textbf{(2.11)}$$

Koeficienty λ_j a γ_k jsou potom určeny z interpolačních podmínek a omezení (2.11) a vedou k následujícímu symetrickému lineárnímu systému:

$$\left[\begin{array}{c|c} \mathbf{A} & \mathbf{P} \\ \hline \mathbf{P}^{\mathbf{T}} & \mathbf{0} \end{array}\right]\left[\begin{array}{c} \boldsymbol{\lambda} \\ \hline \boldsymbol{\gamma} \end{array}\right] = \left[\begin{array}{c} \mathbf{h} \\ \hline \mathbf{0} \end{array}\right]. \tag{2.12}$$

A je matice shodná s maticí v rovnici (2.9) a **P** je $n \times M$ matice se vstupy $p_k(\underline{x}_j)$ pro j = 1,…,n a k = 1,…,M.
Hodnota m použitá v definici výše pochází z Micchelliho rozšíření Věty 2 [Micchelli86].

Věta 3. (Micchelli [Micchelli86]) Nechť $\psi(r) = \phi(\sqrt{r}) \in C^0[0,\infty), \psi^{m+1}(r)$ je ryze monotónní ne však konstantní na $(0,\infty)$ pro $m \geq 0$. Potom pro libovolnou množinu n rozdílných bodů $\{\underline{x}_j\}_{j=1}^n$ splňuje podmínku hodnosti matice **P**, hod(**P**) = M, kde P je $n \times M$ matice v rovnici (2.12) . Plná matice $(n+M)\times(n+M)$ v rovnici (2.12) je regulární. Mimo to je $\overline{m}$ nejmenší m takové, že $\psi^{m+1}(r)$ je ryze monotónní, potom pro libovolný nenulový vektor $\boldsymbol{\alpha} \in R^n$ splňuje podmínku $\mathbf{P}^{\mathbf{T}}\boldsymbol{\alpha} = \mathbf{0}$ a platí následujíc relace: $(-1)^{\overline{m}+1}\boldsymbol{\alpha}^{\mathbf{T}}\mathbf{A}\boldsymbol{\alpha} > 0$, kde **A** je submatice řádu n (2.12) .

Tato věta zajišťuje potřebné podmínky pro jednoznačnou řešitelnost rozšířené RBF metody. Můžeme například použít tuto větu abychom ukázali, že rozšířená RBF metoda je jednoznačně řešitelná pro kubické a TPS bázové funkce pokud m = 1 a podmínky zdrojových bodů dat $\{\underline{x}_j\}_{j=1}^n$ jsou splněny. Pokud je tedy m = 1, tak je RBF interpolant rozšířen o konstantu a lineární polynom. Například ve dvou dimenzích je RBF interpolant určen jako:

$$f(x,y) = \sum_{j=1}^{n} \lambda_j \phi\left(\sqrt{(x-x_j)^2 + (y-y_j)^2}\right) + \gamma_0 + \gamma_1 x + \gamma_2 y. \tag{2.13}$$

Pokud se znovu vrátíme k příkladu s TPS RBF, který jsme uvedli výše, kdy všechny body byly vrcholy jednotkového simplexu a matice A byla tedy singulární, tak toto již není možné s rozšířenou RBF metodou. Nastane totiž situace, kdy část RBF interpolantu v rovnici (2.10) bude nulová a rozšiřující konstanta a lineární mnohočlen budou interpolovat data.

Z věty uvedené výše můžeme vyvodit několik základních předpokladů. Je vidět, že je rozšířená RBF metoda mnohem více volnější k volbě bázové funkce $\phi(r)$, která může být použita, než základní RBF metoda. Zároveň ale klade mnohem větší nároky na datové body $\{\underline{x}_j\}_{j=1}^n$, jež mohou být použity. Jediná podmínka omezující datové body u základní RBF metody je jejich rozdílnost (v datové množině nejsou dva shodné body). Na rozdíl od rozšířené RBF metody, která vede datové body k tomu, aby matice P splňovala podmínku hod(**P**) = M. To je stejné, jako bychom řekli, že musí splňovat podmínku (2.11).

Z definice ryze monotónní funkce (Definice 3) víme, že pokud $\psi^{m+1}(r)$ je ryze monotónní, tak je ryze monotónní $\psi^{m+\kappa}(r)$ pro κ = 2, 3, … . Můžeme tedy z věty 3. udělat závěr, že rozšířená RBF metoda je jednoznačně řešitelná pro všechny hodnoty $m \geq \bar{m}$ za předpokladu, že jsou splněny podmínky pro datové body $\left\{\underline{x}_j\right\}_{j=1}^{n}$.

Zvětšováním hodnoty *m* pro rozšířenou RBF metodu překračující minimální hodnotu $\bar{m}$ (například přidáním polynomu vyššího stupně než je minimální stupeň potřebný k dosažení jednoznačné řešitelnosti) se může zdát bezvýznamné poněvadž podmínky kladené na datové body jsou více restriktivní. Přesto má tato technika praktické použití. Například může být použita k tomu, aby RBF interpolant (2.10) kopíroval polynom určitého stupně čehož může být využito k zvětšení přesnosti interpolace pomocí RBF.

Pro rovnici (2.13) dostává lineární systém (2.12) následující tvar

$$\left[\begin{array}{cccc|ccc} \phi_{11} & \phi_{12} & \cdots & \phi_{1n} & x_1 & y_1 & 1 \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2n} & x_2 & y_2 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \phi_{n1} & \phi_{n2} & \cdots & \phi_{nn} & x_n & y_n & 1 \\ \hline x_1 & x_2 & \cdots & x_n & 0 & 0 & 0 \\ y_1 & y_2 & \cdots & y_n & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{array}\right]\left[\begin{array}{c} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \\ \hline \gamma_1 \\ \gamma_2 \\ \gamma_0 \end{array}\right] = \left[\begin{array}{c} h_1 \\ h_2 \\ \vdots \\ h_n \\ \hline 0 \\ 0 \\ 0 \end{array}\right], \tag{2.14}$$

který budeme následně přepisovat do podoby

$$\mathbf{B}\left[\begin{array}{c} \boldsymbol{\lambda} \\ \hline \boldsymbol{\gamma} \end{array}\right] = \left[\begin{array}{c} \mathbf{h} \\ \hline \mathbf{0} \end{array}\right]. \tag{2.15}$$

Věty 1,2 a 3 nám zajistili, že RBF metoda má řešení jak ve své základní formě pro některé bázové funkce, tak v rozšířené formě. RBF metoda tedy může být použita k řešení interpolace dat.

2.4 Řešení a řešitelnost lineárního sytému

Než budou uvedeny podrobnosti o metodách pro řešení lineárního systému (2.15) je nutné se zmínit o řešitelnosti tohoto lineárního systému.

Nechť je dáno, že

$$f(x_j, y_j, z_j) \equiv f\left(\underline{x}\right) = 0, \qquad j = 1, \ldots, n \tag{2.16}$$

pro všechny body, které jež budou interpolovány. Takto způsobem bylo nadefinováno tzv. implicitní vyjádření trojrozměrného objektu definovaného množinou bodů $\left\{\underline{x}_j\right\}_{j=1}^{n}$, kterým se standardně provádí zápis objektů, jejichž povrch je potřeba nalézt pomocí RBF metody [Turk02, Turk99, Carr01].

Bude-li zvolena bázovou funkci ϕ , tak bude možné vypočítat všechny hodnoty matic **B** (2.15) . Z rovnice (2.16) plyne, že $h_j = 0$ a tedy vektor **h** bude nulový vektor. Lze tedy vytvořit homogenní systém (2.17) , ze kterého je možné vypočítat hodnoty vektorů λ a γ.

$$\mathbf{B}\begin{bmatrix}\boldsymbol{\lambda}\\ \hline \boldsymbol{\gamma}\end{bmatrix} = \begin{bmatrix}\mathbf{0}\\ \hline \mathbf{0}\end{bmatrix} \tag{2.17}$$

Tento systém obsahuje n (n = n + M + 1) homogenních rovnic, které mají vždy nulové řešení **0** = (0,0,…,0). Jestliže matice homogenního systému má hodnost *h*, tak má systém (*n*-*h*) lineárně nezávislých řešení a každé řešení tohoto systému je lineární kombinací těchto (*n*-*h*) řešení. Pokud *h* = *n*, tak systém má pouze triviální řešení **0** = (0,0,…,0). Pokud *m* = *n* (počet řádek je roven počtu sloupců) tak lineární systém má řešení tehdy a jen tehdy pokud determinant systému je roven nule. Matice lineárního systému **B** má na diagonále nulové prvky a mimo diagonálu nenulové. Může být potom dokázáno, že řád lineárního systému je roven *n* a jeho determinant je nenulový.

Definice 6. Čtvercová matice $\mathbf{A} = (a_{ij})$ n-tého řádu se nazývá regulární, je-li její determinant $|a_{ij}|$ různý od nuly (tj. **A** má hodnost n). [Rektorys95].

Lze konstatovat, že uvedený lineární systém má pouze jedno řešení, nulové (triviální) řešení. Toto přesné řešení systému je však nepotřebné, neboť nemůže být dále použito pro zobrazovací metody. Pro zobrazovací metody je zapotřebí aproximace takového řešení. K získání aproximace řešení je nutné přidat do vstupních dat další hodnoty jež vedou k získání nenulového řešení. Dodatečné hodnoty jsou tzv. perturbace. Perturbace přidají do systému nové rovnice a systém tedy není dále homogenní. Podrobněji se o perturbacích zmíním v části věnované interpolace 3D dat.

Velmi populární metodou na řešení systému lineárních rovnic definovaného rovnicí (2.14) se v souvislosti s RBF metodou stala LU faktorizace [Turk99, Turk01, Turk02, Morse01] nebo [Yngve02]. Je to dáno hlavně její jednoduchostí. Tato metoda vychází z pravidla, že **LU** = **A**. Matice systému je dekomponována na horní (**U** matice) a dolní trojúhelníkovou (**L** matice) matici s jedničkami na diagonále. Dekompozici a následné řešení lze zapsat:

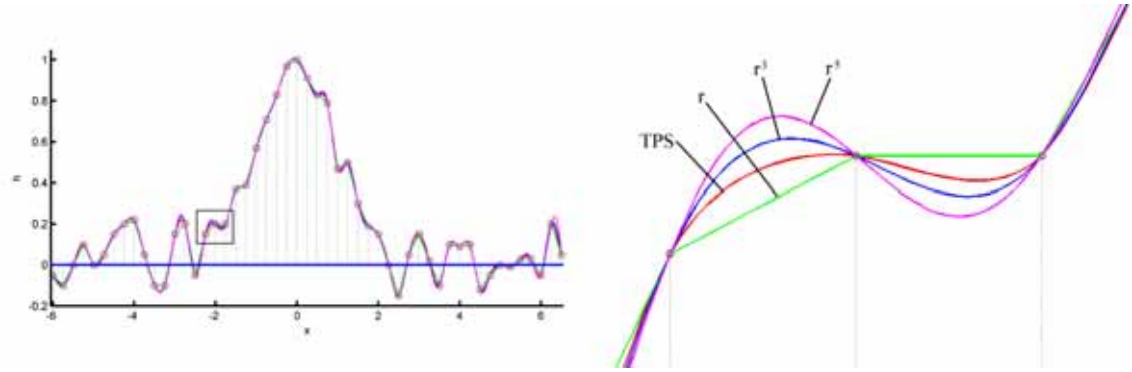
$$\begin{aligned}&\mathbf{Ax} = \mathbf{LUx} = \mathbf{b}\\ &\mathbf{Ly} = \mathbf{b}\\ &\mathbf{Ux} = \mathbf{y}.\end{aligned} \tag{2.18}$$

Velikou výhodou LU faktorizace je možnost jednoduše řešit více lineárních systému se stejnou maticí **A**, ale rozdílnou pravou stranou (vektor **b**). Faktorizace může být vyřešena bez znalosti pravé strany a je k tomu potřeba n^3 operací. Pokud je známa pravá strana tak je potřeba pouze n^2 operací. Mohou být také použity jiné iterační metody jako například Choleskyho faktorizace [Beatson00] nebo GMRES [Beatson99].

2.5 Bázové funkce

Volba bázové funkce ϕ má vliv na tvar výsledné interpolace vstupních dat. V literatuře existuje mnoho různých bázových funkcí. Bázové funkce mohou být rozděleny do dvou skupin a to po částech hladké a nekonečně hladké. Mezi po částech hladké funkce patří

například lineární, kubická, *quintic* nebo *thin plate spline* funkce. Hladké bázové funkce jsou například Gaussovská, inverzní kvadratická , inverzní *multiquadric* nebo *multiquadric*.



Obrázek 1: Interpolace dat různými po částech hladkými bázovými funkcemi. (vlevo – interpolace všech hodnot, vpravo – výřez a označení bázových funkcí)

Pro po částech hladké funkce platí, že se zvětšujícím se počtem bodů konverguje RBF interpolant algebraicky k dostatečně hladké funkci. Míra konvergence je přímo úměrná hladkosti bázové funkce a míra se často zvětšuje se zvětšující se dimenzí. Hladkost bázové funkce má vliv také na stabilitu lineárního systému (2.12). Se zvětšujícím se počtem bodů ve stanovené oblasti se algebraicky zvětšuje i číslo podmíněnosti matice lineárního systému s mírou vztahující se k nepravidelnosti $\phi(r)$ v počátku [Fornberg02, Schaback94].

Typ bázové funkce	$\phi(r), r \geq 0$
Gaussova (GRBF)	$e^{-(\varepsilon r)^2}$, $e^{\frac{-r^2}{2\varepsilon^2}}$
Inverzní kvadratická (IQ)	$\frac{1}{1+(\varepsilon r)^2}$
Inverzní multiquadric (IMQ)	$\frac{1}{\sqrt{1+(\varepsilon r)^2}}$
Multiquadric (MQ)	$\sqrt{1+(\varepsilon r)^2}$
Lineární	r
Kubická	r^3
Thin-plate spline (TPS)	$r^2 \log r$
Quantic	r^5

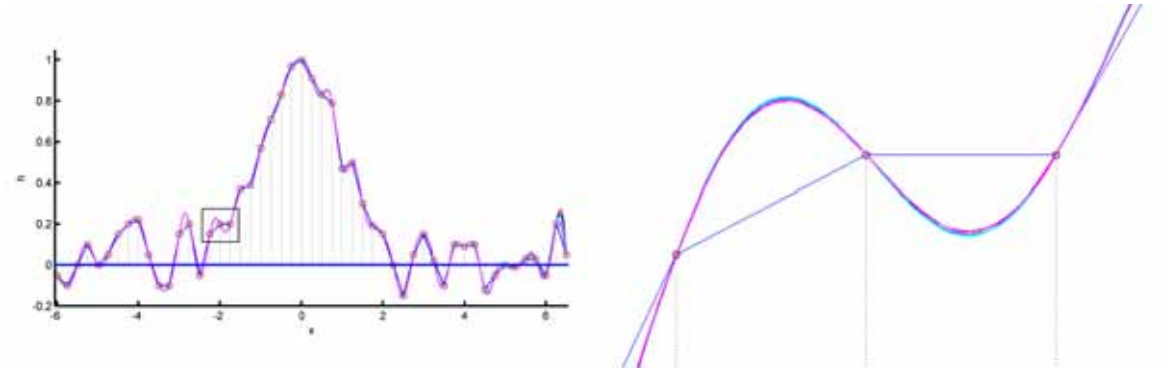
Tabulka 1. Základní typy bázových funkcí.

Mezi nejznámější a nejvíce používaně hladké funkce patří *thin-plate spline* bázová funkce, kterou představil Duchon [Duchon77] a Micchelli [Micchelli86] definoval podmínky jednoznačné řešitelnosti lineárního systému využívajícího TPS bázovou funkci. TPS bázová funkce má fyzikální opodstatnění a je uváděna pro interpolace ve 2d [Morse01, Turk99].

Ekvivalentem TPS bázové funkce pro 3D je uváděna kubická bázová funkce. Rozšířením RBF metody o dodatečné podmínky získáme přirozený kubický spline [Fornberg02]. Stejně je tomu pro *quantic* bázovou funkci, která má spojitost s přirozeným

spine pátého řádu. Kvadratické bázové funkce a vůbec bázové funkce r^{2k}, $k = 1,\ldots,n$ se nepoužívají [Powell92].

Klasická lineární bázová funkce byla prvním pokusem Hardyho [Hardy90]. Použití této bázové funkce v podstatě znamená interpolaci dat lineární kombinací posunující se absolutní hodnoty bázové funkce, kde vrcholem každé bázové funkce je jeden ze zdrojových bodů. Tato funkce se příliš nehodila pro interpolaci, proto ji rozšířil tak, aby byla bázová funkce spojitě diferencovatelná a odvodil tak MQ bázovou funkci.



Obrázek 2: Interpolace dat různými hladkými bázovými funkcemi.
(vlevo – interpolace všech hodnot, vpravo – výřez)

Na rozdíl od po částech hladkých bázových funkcích, přesnost a stabilita pro nekonečně hladké bázové funkce závisí na počtu datových bodů a hodnotě parametru ε. Pokud je konstantní parametr ε a zvětšuje se počet datových bodů, tak RBF interpolant konverguje k dostatečně hladké funkci spektrálně [Madych92b]. V určitých speciálních případech jako například GRBF, tak se RBF interpolant konverguje „super-spektrálně“. Madych [Madych92a] ukázal, že pro konstantní počet bodů může být přesnost často zlepšena zmenšením parametru ε. Neboť zmenšení parametru ε nebo zvětšení počtu bodů má závažný vliv na stabilitu lineárního systému (2.12) . Pro konstantní ε, číslo podmíněnosti matice lineárního systému roste exponenciálně s počtem přibývajících bodů. Pro konstantní počet datových bodů dochází ke zvětšování s $\varepsilon \to 0$. Parametr ε je někdy nazýván „tvarovací parametr“.

Jak bylo uvedeno, tak k základním nekonečně hladkým bázovým funkcím patří MQ bázová funkce, kterou představil Hardy [Hardy71] a použil ji na interpolaci topologických dat. Její výhodou je, že i v základním tvaru RBF metody poskytuje jednoznačná řešení. MQ RBF jsou nekonečně hladké a s volbou parametru ε se chovají tak, že malé ε vede k dobré přesnosti a velké ε zabezpečuje dobrou podmíněnost [Fornberg02]. Hardy postupně doplnil IMQ a IQ bázové funkce [Hardy90].

Další možnou volbou je bázové funkce Gaussova typu (GRBF). Existuje několik různých variant GRBF bázové funkce. Jednou z nich je například variant uvedená v tabulce 1, první řádek vpravo, kterou použil Schagen [Schagen79] pro kreslení kontur geodetických dat. Schagenova GRBF má definovanou hodnotu tvarovacího parametru jako korelační vzdálenost. Podrobnější analýzou Gaussovské bázové funkce se zabývá například [Platte05, Schoenberg38, Powell92].

Obrázek 2 ukazuje interpolaci hodnot hladkými funkcemi. Jelikož každá z nich má tvarovací parametr, tak je nutné si ho uvést, aby byla kompletní představa o možnostech těchto bázových funkcí. Pro MQ je $\varepsilon = 1.0$, pro IQ je $\varepsilon = 1.0$, pro IMQ je $\varepsilon = 1.0$, pro GRBF v Tabulce 1 vlevo je $\varepsilon = 5.0$ a nakonec pro Schagenovu GRBF je $\varepsilon = 0.25$. Ještě je

nutné podotknout, že hodnota tvarovacího parametru je závislá na rozložení dat. Velmi znatelné je to u Schagenovi GRBF.

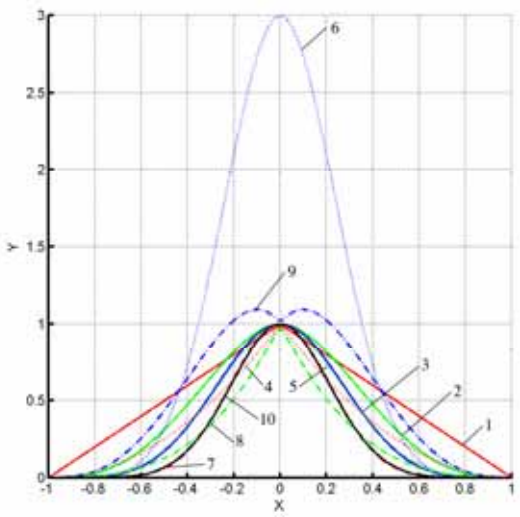
Mezi zajímavé a v posledních letech velmi používané bázové funkce se řadí „*Compactly-Supported*“ radiální funkce (CSRBF). V roce 1995 je představil Wu [Wu95] jako kompaktní, lokální bázové funkce, které garantují pozitivní definitnost matice A (2.12) . Používají principu lokality známého u B-splinu což znamená, že změna jednoho bodu vyvolá pouze lokální změnu interpolační funkce. Wu definoval podmínky pro pozitivní definitnost CSRBF a odvození bázových funkcí vyšší spojitosti. Z jeho práce následně Wendland [Wendland95] odvodil, na základě mocninné funkce, méně výpočetně náročně CSRBF. Jeho funkce mají následující formu:

$$\phi(r) = \begin{cases} (1-r)^p P(r) & 0 \le r \le 1 \\ 0 & \mathrm{r} > 1 \end{cases} \qquad \textbf{(2.19)}$$

Kde $P(r)$ je polynom jehož odvození můžete nalézt v [Wendland95]. Funkce mají různé stupně spojitosti C^k a jsou určeny pro různé dimenze *d*. Následující tabulka představuje radiální bázové funkce, které představil Wendland.

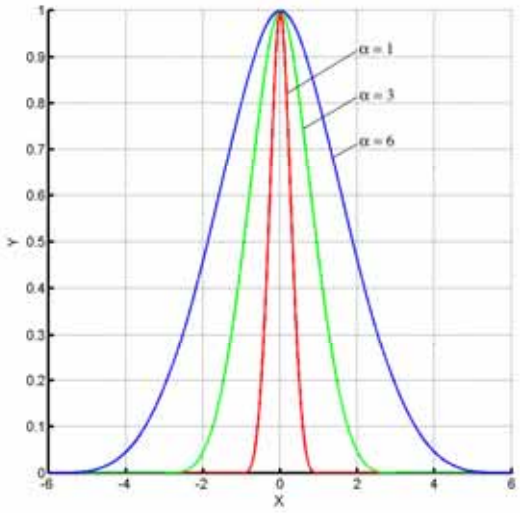
id	dimenze	funkce	spojitost	Obrázek 3
1	d = 1	$(1-r)_+$	C^0	červená – plná
2		$(1-r)_+^3(3r+1)$	C^2	zelená – plná
3		$(1-r)_+^5(8r^2+5r+1)$	C^4	modrá – plná
4	d = 3	$(1-r)_+^2$	C^0	červená – tečkovaná
5		$(1-r)_+^4(4r+1)$	C^2	zelená – tečkovaná
6		$(1-r)_+^6(35r^2+18r+3)$	C^6	modrá – tečkovaná
7		$(1-r)_+^8(32r^3+25r^2+8r+1)$	C^6	červená – čerchovaná
8	d = 5	$(1-r)_+^3$	C^0	zelená – čerchovaná
9		$(1-r)_+^3(5r+1)$	C^2	modrá – čerchovaná
10		$(1-r)_+^7(16r^2+7r+1)$	C^4	černá – plná

Tabulka 2. Compactly-Supported radiální funkce [Wendland95].



Obrázek 3: Průběh CSRBF funkcí uvedených v Tabulka 2.

Uvedené funkce mají poloměr působnosti (*radius of support*) roven 1, tedy na intervalu <-1, 1>. V ostatních případech můžeme upravit působnost bázové funkce parametrem α takto $\phi(r/\alpha)$. Tímto získáme funkci s poloměrem působnosti α (Obrázek 4).



Obrázek 4: Rozdílné hodnoty parametru α pro CSRBF.

Volba poloměru, ve kterém má zvolená bázová funkce vliv je velmi důležitý aspekt ovlivňující výsledek po použití CSRBF. Pokud bude totiž rádius příliš malý, tak se může stát, že bázová funkce nedokáže zahrnout veškerá omezení vstupní množiny bodů, jakým jsou například díry. Naopak příliš velký rádius bude mít nepříznivý vliv na řídkost matice **A** (2.12) a zvětší se tak výpočetní náročnost lineárního sytému (2.12) . Velkým poloměrem tedy degradujeme výhodu CSRBF a zjednodušeně řečeno říká, že body umístěné za poloměrem působnosti funkce už nemají na aktuální bod žádný vliv.

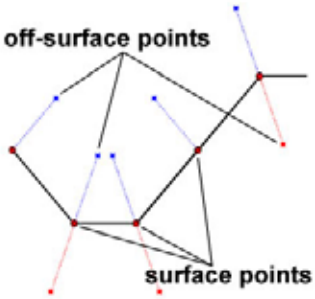
Výhodou klasických radiálních bázových funkcí Gaussova typu, TPS nebo MQ proti CSRBF je jejich jednoduchá reprezentace, která platí pro libovolnou dimenzi prostoru *d*. Je to důsledek Schoenbergova teorému [Schoenberg38]. Proto jsou definovány různé CSRBF pro různé dimenze (Tabulka 2).

2.6 Použití RBF metody

Hlavní oblast kde se využívá RBF metoda je rekonstrukce 3D objektů zadaných pouze body definujících povrch, tzv. *on-surface* body, popřípadě s nějakou dodatečnou informací. Pro takto zadaný objekt se provádí tzv. implicitní popis objektu (2.16) . Z bodů objektu je pak vytvořen lineární systém (2.14) , který je pak řešen. Jak již bylo zmíněno, tak řešením systému, kde je na pravé straně nulový vektor bychom obdrželi pouze přesné triviální řešení, které není možné dál použít pro vizualizaci povrchu objektu. Jsou proto nutné přidat perturbace do systému, které umožní získání nenulového řešení. V případě 3D objektu se po do základní množiny bodů přidávají ještě další body s nenulovou hodnotou. Tyto body jsou nazývány jako tzv. *off-surface* body. Dodefinováním těchto bodů dostáváme obvyklejší interpolační problém: Nalezněte *f* takové, že

$$\begin{aligned} f(x_j, y_j, z_j) &= 0, \qquad j = 1, \ldots, n \\ f(x_j, y_j, z_j) &= d_j \neq 0, \qquad j = n+1, \ldots, N. \end{aligned} \tag{2.20}$$

Tímto však vzniká nový problém jak generovat tyto *off-surface* body $\{(x_j, y_j, z_j)\}_{j=n+1}^{N}$ a jejich hodnoty d_i. Obvyklá volba funkce popisující způsob přidání hodnot d_i funkci f je tzv. *signed-distance* funkce neboli znaménková funkce. Hodnoty d_j potom značí vzdálenost k nejbližšímu bodu ze vstupní množiny bodů. Znaménko hodnoty d_j se potom volí podle polohy bodu. Pokud je bod uvnitř objektu, tak je hodnota záporná a pokud je bod vně objektu, tak je kladná (nebo naopak). Standardně se dodatečné body generují ve směru normály ve vrcholu nebo normály povrchu [Turk02]. Jak je ukázáno na Obrázek 5, tak dodatečné body nemusí být na obou stranách všechny.



Obrázek 5: Ukázka generování dodatečných bodů ve směru normály.

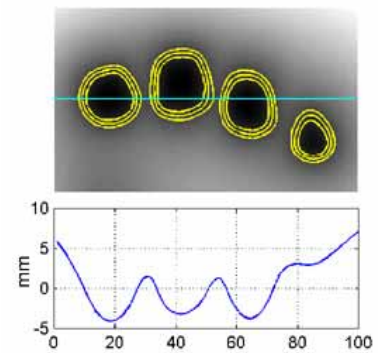
Zkušenosti ukazují, že je lepší přidat body na obou stranách povrchu. Příklad můžete vidět na Obrázek 6, kde je ukázka rekonstrukce ruky získané z laserového skeneru. Body které mají zelenou barvu jsou originální bodu získané skenerem a body ostatních barev jsou *off-surface* body, kde jejich barva udává vzdálenost od originálních bodů. Červené body mají kladnou hodnotu a modré body mají zápornou hodnotu. Z obrázku je také vidět, kde jsou problémy při takové rekonstrukci. První problém je odhad normály, v jejímž směru budou přidávány *off-surface* body a druhým problémem je určení vzdálenosti v jaké budou.

Obrázek 6: Rekonstrukce ruky definované množinou bodů se správnou (b) a špatnou (c) definicí dodatečných bodu. [Carr01]

Pokud by byla nad vrcholy objektu definována trojúhelníková síť, tak by bylo velice jednoduché vypočtení normály ve vrcholu každého vrcholu, neboť trojúhelníková síť poskytuje informaci o sousednosti vrcholů. Jestliže však tato informace není, což je v případě shluku bodů běžné, tak je nutné odhadnout normálu z okolí vrcholu. Je potřeba odhadnout nejen směr normály, ale také její význam, což znamená určit kde je uvnitř a kde je venku. Jednou z možností je lokální aproximace bodů rovinou a odhadnutí směru normály, avšak toto pravidlo není možné dodržet u všech objektů. Je nutné zachovat konzistentnost znaménka na stranách objektu. V každém případě je vhodné mít nějakou dodatečnou informaci jakou je například poloha scanneru. Obecně je velký problém

odhadnou normály kdekoli, a proto pokud je směr nebo význam normály v bodě nejasný, tak se s normálou nepracuje. Lepší volbou je potom ponechání pouze základní informace o povrchu v daném bodě. Jedna z možných metod na nalezení normál je popsána v [Hoppe92].

Podaří-li se získat normály ve vrcholech, tak je možné začít generovat *off-surface* body ve směru normál. Při procesu generování off-surface bodů je nutné zajistit, aby vektory ve směru normál neprotínaly ostatní části povrchu. *Off-surface* body jsou pak přidávány tak, že nejbližším bodem k tomuto bodu je bod na povrchu objektu, ze kterého byl bod generován. Při dodržení této podmínky je rekonstruovaný povrch necitlivý na projekční vzdálenost $|d_j|$. Obrázek 6c znázorňuje problém, který nastane pokud jsou body generovány v nevhodné vzdálenosti ve směru normály. *Off-surface* body byly generovány v konstantní vzdálenosti od povrchu. Výsledný povrch, kde f je nula, zdeformovaný v okolí prstů, kde došlo k protnutí opačných normál a generování off-surface bodů ve špatné vzdálenosti od povrchu. Na Obrázek 6a a 6b byla provedena kontrola vzdálenosti *off-surface* bodů a dynamická projekce pomohla zajistit generování bodů ve vhodné vzdálenosti od povrchových dat.



Obrázek 7: Průřez prsty ruky rekonstruované ze shluku bodů na Obrázek 6. [Carr01]

Obrázek 7 znázorňuje průřez prsty ruky. Je zde možné vidět jak RBF funkce aproximuje distanční funkci v blízkosti povrchu objektu. Na horní části obrázku jsou vidět kontury s hodnotou +1, 0 a -1. Spodní část obrázku pak ukazuje průběh funkce v řezu označeného čárou na horní části obrázku. Vidíme jak *off-surface* body generují funkci, která má hodnotu gradientu blízkou 1 u povrchu. Povrch odpovídá průsečíku funkce s nulovou hodnotou.

2.7 Urychlovací techniky

RBF metoda je velmi výpočetně náročná, proto existuje mnoho různých způsobů, jak metodu urychlit. Některé z nich jsou založeny na předzpracování, jako například redukce vstupní množiny bodů a jiné na urychlení řešený lineárního systému. V této části budou některé z nich představeny.

2.7.1 Fast Multipole Method

Fast Multipole Method (FMM) byla představena v [Greengard87] a pro urychlení RBF metody ji použil Carr [Carr01]. FMM metoda byla určena pro rychlé vyhodnocení harmonických RBF ve 2D a 3D [Beatson97]. Metoda je založená na zmenšení přesnosti výpočtu a tudíž pouze aproximuje daná data. Základem metody je určení přesnosti výpočtu, která je rozhodující pro kvalitu výsledné aproximace. Dále je prostor hierarchicky rozdělen na části se shodným počtem bodů. Na jednotlivých částech prostoru je provedena

aproximace s definovanou chybou. V posledním kroku je provedeno zpřesnění dle požadované přesnosti. Metoda značně sníží výpočetní čas v porovnání s přímým vyhodnocením. Pro metodu je potřeba $n \cdot \log(n)$ operací, ale implementace metody je velice náročná. Podrobnější popis metody naleznete například v [Beatson97].

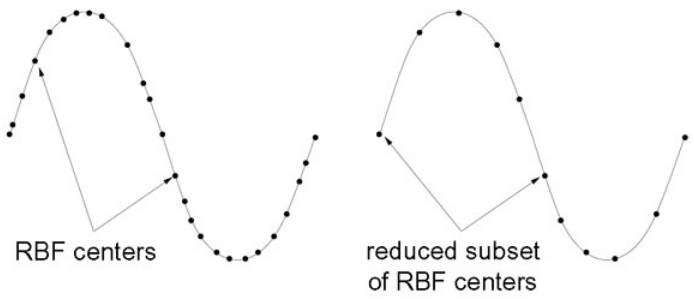
2.7.2 Redukce bodů

Základní RBF metoda používá všechny body ze vstupní množiny bodů. To může být v případě modelů s velkým množstvím bodů problém, neboť vyhodnocení RBF metody v jedno milionu vrcholů znamená výpočet lineárního systému s jedním milionem rovnic. Další používanou urychlovací metodou je tedy redukce vstupní množiny bodů tak, aby redukce měla co nejmenší vliv na kvalitu výsledné aproximace (Obrázek 8). Může zde být použit algoritmus, který postupně prokládá množinou bodů s definovanou přesností.

Jednoduchý algoritmus má následující kroky:

1. Výběr podmnožiny vrcholů z množiny všech N vrcholů a proložení této podmnožiny RBF metodou.
2. Vyhodnocení rozdílu, $\varepsilon_j = f_j - f(\underline{x}_j)$, ve všech vrcholech.
3. Jestliže $\max\{|\varepsilon_j|\} <$ *požadovaná přesnost* => konec.
4. Jinak přidej nové vrcholy z množiny N tam, kde ε_j je velká.
5. Prolož RBF metodou => bod 2.

Jestliže je určená rozdílná přesnost ε_j v každém bodě, tak může být podmínka ve 3. kroku nahrazena podmínkou $|\varepsilon_j| < \delta_j$.



Obrázek 8: Ukázka redukce počáteční množiny bodů. [Carr01]

Redukce počtu bodů není podstatná, pokud je použita FMM metoda popsaná výše. Použití metody redukce počtu vstupních bodů naleznete v [Carr01]. Jeden z možných způsobů redukce vstupní množiny bodů může být i redukce trojúhelníkové sítě objektu [Franc02]. Tato metoda může být použita pouze v případě, že je na počátku známa trojúhelníková síť objektu. Tento způsob může být vhodný například pokud chceme popsat model RBF metodou pro pozdější zpracování s definovanou přesností.

2.7.3 Výběr podmnožiny bodů

Výběr podmnožiny bodů z hlavní množiny bodů je používán hlavně společně s CSRBF [Morse01]. Vychází to z vlastností CSRBF, neboť CSRBF mají definovaný poloměr vlivu a bázová funkce $\phi(\underline{x}) = 0$ pro všechna $\underline{x}$ za tímto poloměrem. Velmi používanou metodou

pro nalezení množiny všech bodů, které jsou ve vzdálenosti r od zvoleného bodu $\underline{x}$ je uspořádání dat do stromové struktury zvané k-d strom.

k-d strom je vícerozměrný binární strom s následujícími pravidly pro tvorbu stromu a s bodem $\underline{x}$ jako kořenem a podstromy T_{levy} a T_{pravy}

$$\begin{aligned} \forall y \in T_{levy} : y^d \le x^d \\ \forall y \in T_{pravy} : y^d > x^d \end{aligned}, \tag{2.21}$$

kde se dimenze třídění d mění s každou úrovní stromu. k-d strom může být použit k nalezení všech bodů ve vzdálenosti r v $O(n \log n)$ čase.

Výsledná matice je velmi řídká a pro uložení takovéto matice je potřeba pouze $O(n)$ paměti. Jestliže je matice lineárního systému řídká, tak může být použita metoda na řešení lineárního systému rovnic, která této výhody využije. Výpočetní náročnost metody na řešení řídkého systému lineárních rovnic už potom závisí jak na „řídkosti“ matice, tak na způsobu uložení prvků takového systému a dalších speciálních vlastnostech systému.

2.7.4 Volba metody na řešení systému rovnic

Jedna z možných urychlovacích metod může být také volba metody na řešení lineárního systému rovnic. Nejpoužívanější metodou pro řešení je LU faktorizace. LU faktorizace se používá jak v případě plné matice, tak i v případě řídké matice. Může být také použita na podmnožinu bodů v případě urychlovacích metod uvedených výše. Další metody, které jsou používány jsou například GMRES [Mika96] nebo SVD, kdy je původní maticový systém převeden na systém, který je snadněji řešitelný SVD metodou [Laga02]. Algoritmická složitost těchto metod je zmíněna v dalším paragrafu.

2.8 Algoritmická složitost

Výpočet a použití RBF metody muže být analyzováno ve třech částech:

1. Vytvoření systému lineárních rovnic
2. Řešení systému lineárních rovnic
3. Vyhodnocení interpolační funkce

2.8.1 Vytvoření systému lineárních rovnic

Významný podíl na výpočetní náročnosti RBF metody má vytvoření matice systému lineárních rovnic složenou z $\phi_{ij} = \phi(\| \underline{x}_i - \underline{x}_j \|)$. Jestliže je totiž použita například TPS bázová funkce $\phi(r) = r^2 \log(r)$ nebo bázová funkce $\phi(r) = r^3$, tak matice lineárního systému je skoro nenulová mimo diagonálu. Při použití těchto funkcí je tedy nutné provést výpočet bázové funkce pro všechny vzdálenosti bodů hlavní množiny. Vzhledem k symetričnosti matice klesá výpočetní náročnost na polovinu, ale výpočetní náročnost je stále $O(n^2)$. Mimoto paměť potřebná na uložení takové matice je $O(n^2)$ hodnot datového typu ***double***. Pro reálné objekty (10^6 bodů) může být paměťová náročnost mnohem větším problémem než vytvoření systému. V tomto případě se pak uplatňují metody, kde je možné zpracovávat objekt po částech, speciální bázové funkce (např. CSRBF) nebo různé urychlovací metody.

2.8.2 Řešení systému lineárních rovnic

Přestože nejrozšířenější metodou pro řešení systému lineárních rovnic je LU faktorizace [Turk99] tak tento algoritmus výpočtu, který má složitost $O(n^3)$ může být nahrazen méně výpočetně náročnou iterační metodou. Výpočetní náročnost může být tedy snížena na složitost tvorby lineárního systému. Například iterační metoda GMRES redukuje výpočetní složitost řešení systému lineárních rovnic na O(n log n) operací a O(n) paměti [Beatson99].

2.8.3 Vyhodnocení interpolační funkce

Skoro všechny aplikace nedokáží jednoduše vyřešit váhy RBF metody. K tomu je ještě nutné vyřešit poměrně velké množství funkcí při extrakci iso-plochy, výpočtu normál nebo jiných veličin. Jelikož výrazy $\phi(\| \underline{x} - \underline{x}_j \|)$ v rovnici (2.7) jsou prakticky všechny nenulové pro TPS bázovou funkci s výjimkou kdy $\underline{x} \in \{\underline{x}_j\}$, tak všechny části výrazu musí být použity pro výpočet hodnoty jednoho bodu. Z toho plyne výpočetní složitost při vyhodnocení jednoho bodu, která je $O(n)$.

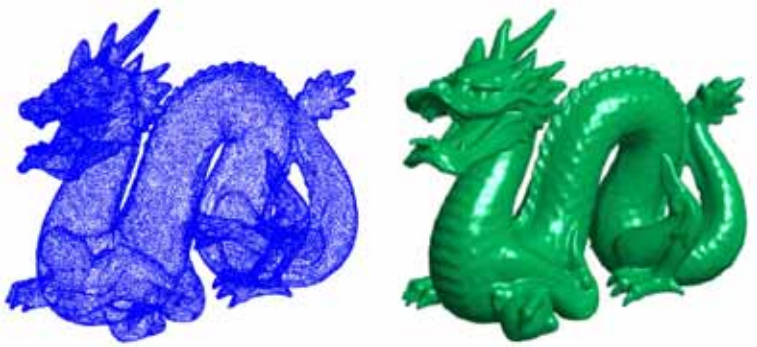
Jestliže je tedy použita TPS bázová funkce, jež minimalizuje energii křivky, tak je nutné počítat s následujícími nevýhodami při výpočtu:

1. $O(n^2)$ výpočtů je potřeba pro vytvoření systému lineárních rovnic.
2. $O(n^2)$ paměti je potřeba (pro skoro plnou matici) k reprezentaci systému.
3. $O(n^3)$ výpočtů je potřeba k vyřešení systému rovnic.
4. $O(n)$ výpočtů je potřeba pro vyhodnocení RBF v jenom bodě.
5. Každý známý bod má vliv na výsledek. To znamená, že při změně jednoho bodu je nutné provést kompletní výpočet znova. Toto je velmi nepříjemná vlastnost při modelování.

2.9 Výhody a nevýhody metody

2.9.1 Výhody

Popis jednou funkcí má mnoho výhod oproti parametrickým plochám a implicitním záplatám. Funkce může být vyhodnocena v libovolném místě při tvorbě trojúhelníkové sítě s libovolnou požadovanou přesností. Řídký, nerovnoměrně navzorkovaný povrch může být popsán RBF a zamezíme tak problému parametrizace povrchu spojený s částečným napojením kubické spline záplaty.



Obrázek 9: Popsání shluku bodů (438,000 bodů) radiální bázovou funkcí. [Carr01]

RBF metoda nabízí kompletní a kompaktní funkcionální popis množiny povrchových dat. Interpolace a extrapolace jsou vlastní funkcionální reprezentaci. RBF asociovaná s povrchem může být vyhodnocena kdekoli k získání trojúhelníkové sítě požadovaného rozlišení. RBF reprezentace je výhodná pro zjednodušování trojúhelníkové sítě a převzorkovací aplikace. Gradient a derivace vyšších řádů jsou určeny analyticky a jsou spojité a hladké v závislosti na volbě bázové funkce. Normály jsou proto spolehlivě spočteny a iso povrchy získané z implicitního RBF modelu jsou manifold. (neprotínají samy sebe).

2.9.2 Nevýhody

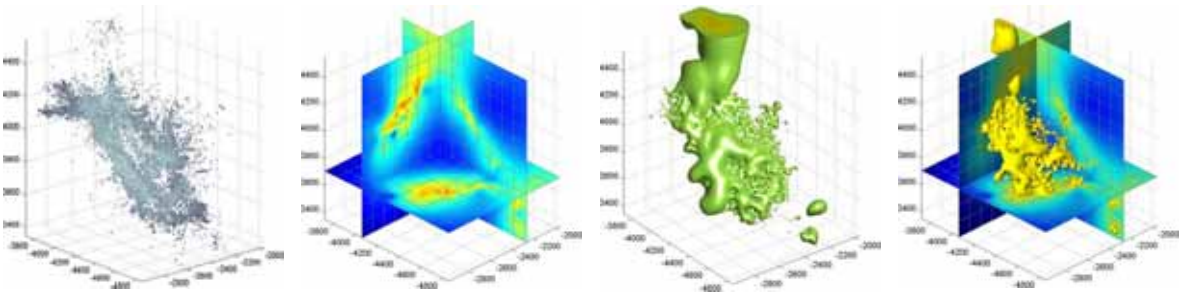
Jestliže RBF metoda je použita na řešení, tak je důležité pro jednoduchou metodu bez jakékoliv redukce vstupní množinu bodů počítat s $O(n^2)$ pamětí a $O(n^3)$ aritmetickými operacemi. Například přímé zpracování draka na Obrázek 9 je potřeba 3,000GB pouze na uložení korespondující matice. Proto přizpůsobení RBF na neskenovaná data reálného objektu není pokládáno za výpočetně uskutečnitelné pro velkou množinu dat.

2.10 Příklady

RBF metoda je použita v mnoha aplikacích. Tato metoda je hlavně použita pro tvorbu implicitního popisu objektu definovaného množinou bodů nebo trojúhelníkovou sítí. Vstupní množina bodů může být z vědeckého měření, počítačové tomografie (CT), magnetické rezonance (MR), 3D skeneru atd.

2.10.1 Vědecké vizualizace

Zobrazovat vědecká data je velmi důležité pro pochopení různých procesů a zobrazení různých skrytých struktur, které nejsou normálně na objektu patrné. Dobrým příkladem můžou být například vizualizace geofyzikálních měření. Obrázek 10 ukazuje datový set obsahující 471031 geofyzikálních měření ve 3D.



a) 3D pohled **b) Hodnoty na rovině** **c) Iso-plocha** **d) Kombinované zobrazení**

Obrázek 10: Vizualizace geofyzikálních dat. [ARANZ]

Obrázek 10*a*, *b*, *c* a *d* ukazují zobrazování dat do více rovin, extrakci iso-plochy a kombinované zobrazení. Toto zobrazení bylo dosaženo pomocí *Fast*RBF toolboxu pro MATLAB. *Fast*RBF toolbox je produktem Applied Research Associates NZ Ltd. [ARANZ].

2.10.2 Vizualizace lékařských dat

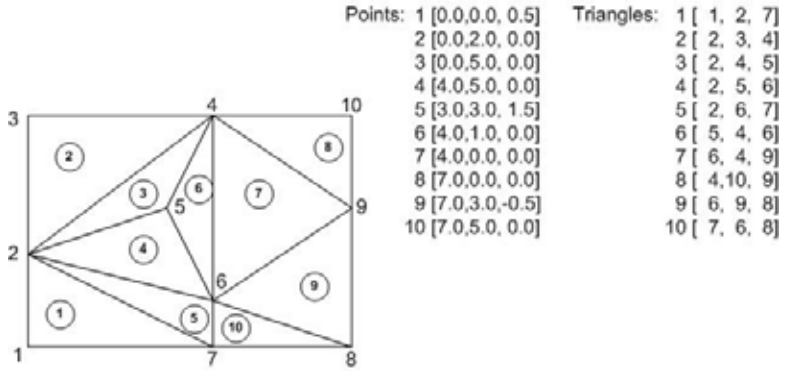
Vizualizace lékařských dat (CT, MR) nebo implantátů můžete nalézt v [Carr97] nebo [Yoo01]. Carr jako jeden z prvních použil RBF metodu na praktický případ rekonstrukce nekompletního povrchu získaného z trojdimenzionálních lékařských dat. Konkrétně použil metodu na návrh lebečních implantátů používaných při poškození, obvykle se jedná o otvor na hlavě. Terry S. Yoo prezentoval rekonstrukci vnitřku cévy z několika endovaskulárních MR obrazů.

2.10.3 Vizualizace naskenovaných 3D dat

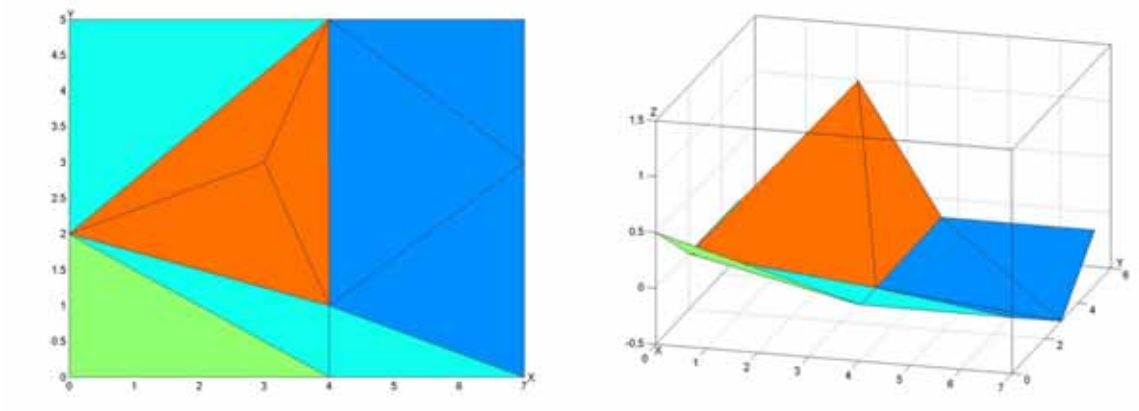
Mnoho příkladů vizualizace naskenovaných 3D dat je v [Carr01]. Obrázek 6 a Obrázek 9 ukazují některé z rekonstruovaných objektů. Data byly získána z 3D scanneru LIDAR nebo CYRAX 2004. Jelikož data mají velmi často velké množství bodů, tak jsou použity různé metody na redukci vstupní množiny bodů (např. FMM).

2.10.4 Příklad vizualizace

V této části bude popsán jednoduchý experiment použití RBF metody pro vizualizaci jednoduchého objektu. Tento experiment poskytne základ pro následující kapitolu. Experiment pomůže pochopit základní použití RBF metody pro interpolaci dat v prostoru. Jsou dány souřadnice několika bodů v prostoru (Obrázek 11). Tyto body definují povrch. Vizualizaci takto definovaného povrchu si můžeme prohlédnout na Obrázek 12.



Obrázek 11: Definice povrchu určeného několika body.

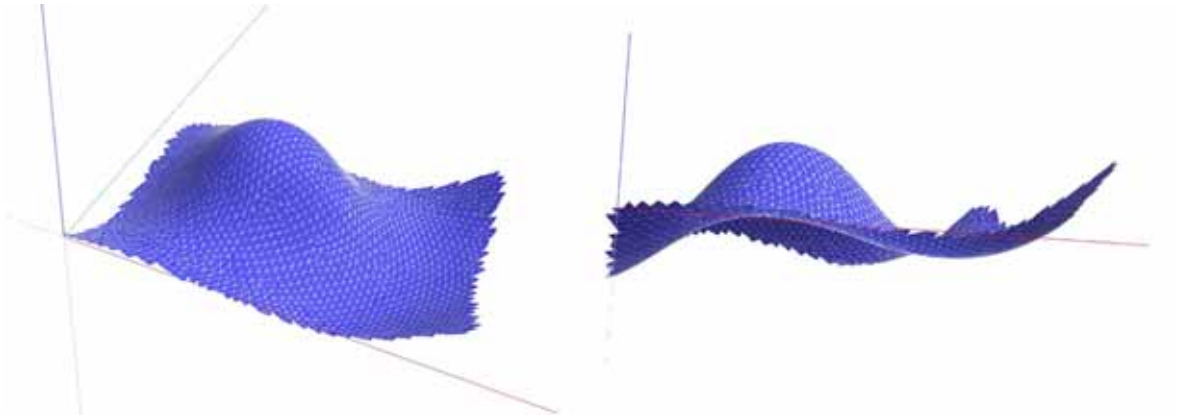


Obrázek 12: Vizualizace povrchu podle definice.

Nyní musíme ještě nadefinovat bodům hodnoty h, která tvoří vektor **h** lineárního systému definovaného rovnicí (2.14) . Jelikož všechny body tvoří jeden povrch, tak musí mít i stejnou hodnotu. Povrch je většinou definován jako nulová iso-hladina a proto všechny vrcholy budou mít hodnotu $h_j = 0$. Pokud sestavíme lineární systém (2.14) , tak bude mít na pravé straně nulový vektor a řešení takového systému rovnic je jen jedno a to je triviální řešení.

Jak již bylo zmíněno dříve, tak musí být do systému přidány další body – perturbace, které nemají na nulovou hodnotu. Tyto body jsou v tomto případě přidávány pro jednoduchost ve směru osy **z**. Body nad původními body budou mít hodnotu 1 a body pod původními body hodnotu -1. Všechny přidané body mají shodnou vzdálenost od původních bodů.

S dodatečnými body je již možné získat netriviální řešení lineárního systému (2.14) a vypočtený vektor λ použít dle rovnice (2.10) pro získání hodnoty kdekoliv v okolí původně zadaných bodů. Tento přístup pak mohou využít zobrazovací techniky jako například *marching cubes* nebo *marching triangles* metoda (Obrázek 13). Ještě musíme podotknout, že v tomto případě byla použita jako bázová funkce TPS funkce.



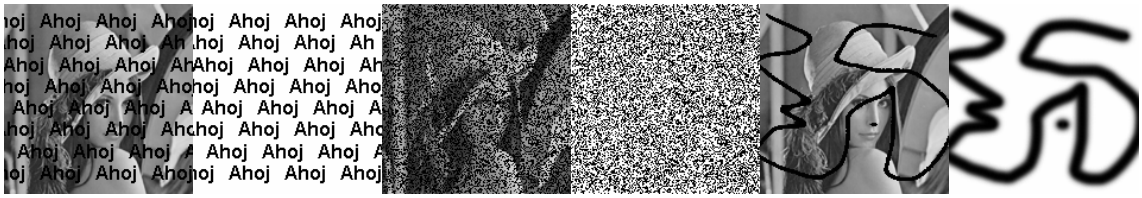
Obrázek 13: Vizualizace povrchu *marching triangles* metodou z rovnice (2.10) a TPS bázovou funkcí.

Tento příklad ukazuje použití RBF metody na interpolaci bodů.

3 Rekonstrukce poškozených obrazů

3.1 Úvod

Existuje mnoho metod zabývajících se rekonstrukcí poškozených obrazů. Hned na začátku však musíme tyto metody rozdělit na metody, které se zabývají odstraňováním šumu a metody opravující obraz poškozený tzv. *inpaintingem*. Slovem inpainting jsou označovány artefakty, které do obrazu nepatří a jeví se jako poškození jako například vepsaný text, šum, škrábance atd. (Obrázek 14) Mezi metody odstraňující šum můžeme zařadit například filtraci obrazu pomocí FFT nebo různé jiné filtry, jejichž podrobnější popis lze nalézt v [Glassner95]. Tyto metody nemohou být použity pro rekonstrukci obrazu poškozeného inpaintingem. Průkopnickou prací v rekonstrukci obrazu poškozeného inpaintingem je označována Bertalmiho metoda založená na parciálních diferenciálních rovnicích [Bertalmio00]. Tato metoda je velmi dobrá a dosahuje zajímavých výsledků (Obrázek 15). Přesto tato metoda nemůže být také použita pro odstraňování šumu a rekonstrukci velmi poškozených obrazů. Metoda využívá propagaci intenzity ze známé části obrazu do poškozené části avšak v případě velkého poškození není zaručena okolní informace. Metoda je plně automatická až na první krok, kdy musí být uživatelem definována oblast inpaintingu. S tímto problémem se ovšem potýkají všechny metody.



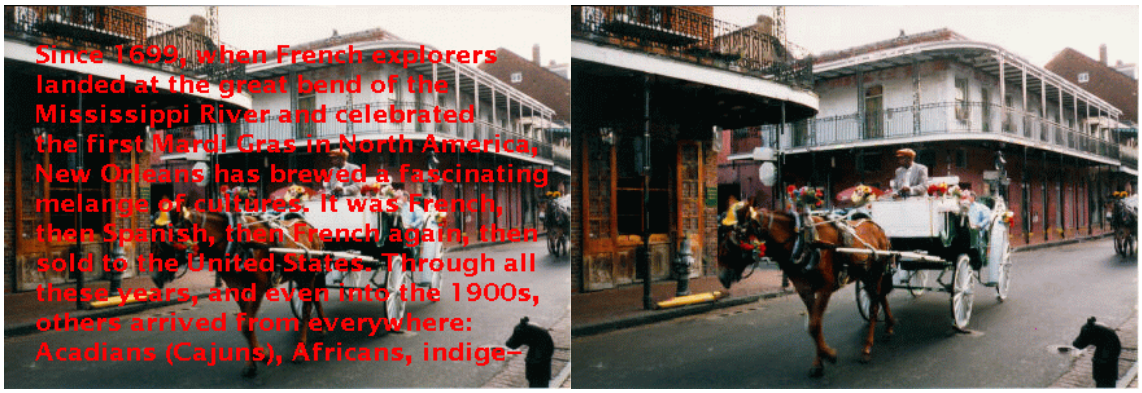
a) b) c)

Obrázek 14: Různé druhy poškození obrazu a) text, b) šum, c) škrábance.

Existují celkem tři skupiny metod rekonstruujících inpainting. Jsou to metody zabývající se opravou filmů, analýzou textury a její použití na poškozená místa a nakonec metody eliminující jednoduchá poškození (*disocclusion* algoritmy). V případě oprav poškození ve filmu se využívá analýzy okolních snímků a doplnění chybějících částí z nich. Tyto techniky nemohou být použity na statický obraz nebo na film, kde je poškozená oblast na mnoha snímcích.

Metody využívající doplňování textury mohou dosahovat velice dobrých výsledků [Hirani96, Sprott04]. Problémem u těchto metod je, že musí být uživatelem určena textura, která se bude přes poškozená místa doplňovat. V případě, že poškozené místo sahá přes množství různých struktur, tak musí být ještě doplněno rozdělení poškozeného místa pro různé textury.

Prvotní práce zabývající se tzv. *Disocclusion* algoritmy je popsána v [Nitzberg93]. Základní myšlenka této metody je v pospojování T-spojem částí obrazu se stejnou úrovní šedé elastickou minimalizující křivkou. Tyto algoritmy jsou určené hlavně pro obrazy s jednoduchými objekty a nemohou být použity na klasické fotografie, se kterými se setkáte dále. Zároveň musí mít oblasti poškození jednoduchou topologii, tzn. bez děr. Poslední metodou, kterou je možné použít pro rekonstrukci poškozeného obrazů je RBF metoda.



Obrázek 15: Poškozený obraz inpaintingem (vlevo) a obraz po rekonstrukci Bertalmiho metodou (vpravo).

3.2 Použití RBF metody ve zpracování obrazu

První zmínky o použití RBF metody ve zpracování obrazu jsou již v [Hardy77, Hardy90], kde Hardy použil svou MQ metodu na rekonstrukci Lincolnova obrazu. Jednalo se vlastně o interpolaci hodnot v obraze. Jeho výsledky byly zajímavé a srovnatelné s metodou založenou na Furierově transformaci a filtrování. Po této první zmínce se RBF metoda ve zpracování obrazu nepoužívala až do doby, kdy Savchenko použil CSRBF [Savchenko02] k získání bodů v poškozené části obrazu. Výhodou CSRBF, které Savchenko použil je získání řídkého lineárního systému a tudíž rychlého řešení. Nevýhodou je, že CSRBF metodu můžeme sice použít na rekonstrukci šumu v obraze avšak neposkytuje příliš kvalitní výsledky.

V následující práci jsme se zaměřili na metodu, která je založena na RBF metodě a která je vhodná jak pro rekonstrukci obrazů poškozených inpaintingem, tak například obrazů poškozených šumem. Dále provedeme srovnání s již existující metodou založenou na CSRBF [Savchenko97] i Bertalmiho metodou [Bertalmio00] a předvedeme výsledky rekonstrukce různě poškozených obrazů.

Obrázek 16: Poškozený obraz (vlevo) a opravený obraz (vpravo) [Savchenko02].

3.3 Definice problému

Nechť máme pravoúhlou oblast Ω definovanou tak, že

$$\Omega = \{(x, y) \mid x = 1, ..., M, y = 1, ..., N\}. \tag{3.1}$$

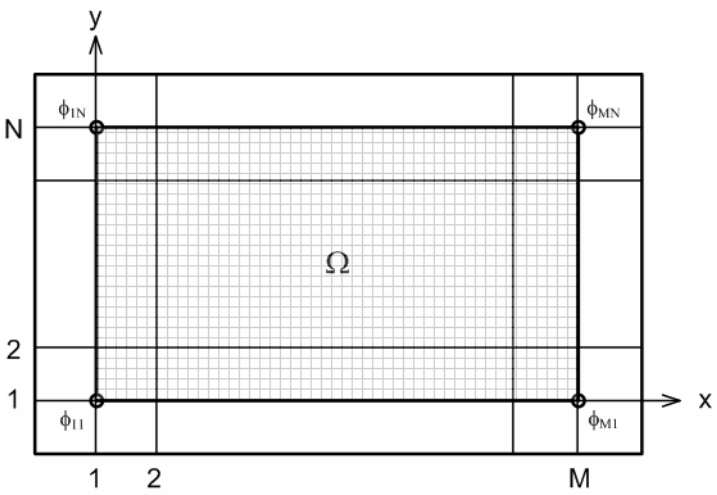
Tato pravoúhlá oblast má tedy rozlišení $M \times N$. Nechť každý bod $p \in \Omega$ má hodnotu h, potom také platí, že

$$p(x,y) = h\,,\ p = \{(x,y,h)\}\,,\ (x,y) \in \Omega\,. \tag{3.2}$$

Předpokládejme, že oblast Ω, kterou budeme dále nazývat „obraz“ je složena ze dvou částí takových, že

$$\Omega = \Omega_c \cup \Omega_i\,,\ \Omega_c \cap \Omega_i = \varnothing\,. \tag{3.3}$$

Ω_c jsou tzv. „správné“ hodnoty, což jsou hodnoty, které do obrazu patří, známe je a budeme s nimi počítat. Ω_i jsou naopak hodnoty „nesprávné“ a tyto hodnoty se budeme snažit nahradit hodnotami správnými. Obraz, který obsahuje pouze správné hodnoty a platí pro něj, že $\Omega \equiv \Omega_c$ je obrazem originálním a tady nepoškozeným. Poškozený obraz je obrazem daným rovnicí (3.3).



Obrázek 17: Definice oblasti Ω ve vztahu k hodnotám matice A ((2.12), (2.14)).

Jestliže označíme originální obraz $\Omega^{\mathbf{A}}$ a poškozený obraz jako $\Omega^{\mathbf{B}}$, pak můžeme také říct, že originální obraz se skládá ze dvou částí tak, že

$$\begin{aligned} \Omega^{\mathbf{A}} &= \Omega_c^1 \cup \Omega_c^2, \Omega_c^1 \cap \Omega_c^2 = \varnothing \\ \Omega^{\mathbf{B}} &= \Omega_c \cup \Omega_i, \Omega_c \cap \Omega_i = \varnothing \end{aligned} \tag{3.4}$$

a pak pokud

$$\Omega_c^1 \equiv \Omega_c, \text{tak } \Omega_c^2 \equiv \Omega_i\,. \tag{3.5}$$

Naším cílem je nalézt takové hodnoty h_i pro všechny $p \in \Omega_i$, aby platilo, že hodnoty h_i jsou „shodně“ s hodnotami h_c z originálního nepoškozeného obrazu, rovnice (3.6) .

$$\begin{aligned} \forall p \in \Omega_i : p &= \{(x, y, h_i)\} \\ \forall p \in \Omega_c^2 : p &= \{(x, y, h_c)\} \\ h_i &\equiv h_c \end{aligned} \tag{3.6}$$

Cílem je tedy zrekonstruovat hodnoty v oblasti Ω_i obrazu Ω. Samozřejmě, že není v naší moci zajistit nalezení „shodných“ hodnot a žádný algoritmus nám to neumožní. Každá metoda rekonstrukce bude nalézat pouze hodnoty přibližné („blízké“ původním hodnotám) a záleží pak již jen na algoritmu, jak se budou tyto nalezené hodny lišit od hodnot původních.

Vstupem dále uvedených metod je většinou poškozený obraz $\Omega^{\mathbf{B}}$ a obraz Ω_i, který budeme nazývat „maska“. To znamená, že víme, jaká část obrazuje poškozená a nezabýváme se tedy metodami detekce poškozené části obrazu. Obraz $\Omega^{\mathbf{A}}$ je použit pouze pro výsledné porovnání rekonstrukce.

V předchozím textu jsme mluvili o rekonstrukci, ale co vlastně budeme rekonstruovat a co znamená hodnota h? Hodnota h v našem případě bude hodnota jasu obrazového bodu p na pozici (x, y). Budeme se tedy zabývat rekonstrukcí jasové hodnoty poškozeného pixelu. Standardně předpokládáme 256 úrovní jasu. V případě černobílého obrazu budeme mít pouze jeden kanál avšak v případě barevného obrazu budeme mít kanály tři, např. R, G a B. Každý kanál bude mít 256 úrovní sytosti.

Obrázek 18: Poškozený černobílý (vlevo) a barevný obraz (vpravo).

3.4 Testované obrazy a masky

Pro porovnání metod budeme používat několik různých obrazů a masek. Obrazy mají rozdílné rozměry a masky zahrnují několik různých druhů poškození. Kompletní přehled všech testovaných obrazů a jejich poškození naleznete v příloze E.

3.5 Metody porovnání rekonstrukce obrazu

Dříve než se budeme zabývat způsoby rekonstrukce obrazu, je nutné definovat, jak budeme rekonstruované obrazy porovnávat a měřit jejich chybu. Metody pro porovnání většinou vycházejí z faktu, že je k dispozici originální nepoškozený obraz, který je nějakým způsobem poškozen, rekonstruován a pak je výsledek po rekonstrukci porovnán s originálním obrazem. Ve zpracování obrazu existuje několik metod jak porovnávat dva obrazy.

Jednou ze základních metod je měření MSE (*mean square error*) neboli střední hodnoty „čtverce“ chyby v jednom obrazovém bodě (pixelu), vzorec (3.8). Jedná se v podstatě o umocněný rozdíl hodnot pixelu v originálním obraze a hodnoty pixelu v rekonstruovaném obraze (3.7). Kde Ω_1 je původní obraz Ω_2 a je obraz rekonstruovaný.

Mocnina zvýrazní chybu. Součet této chyby přes celý obraz a její následné vydělení celkovým počtem pixelů obrazu udává chybu rekonstrukce na jeden obrazový bod.

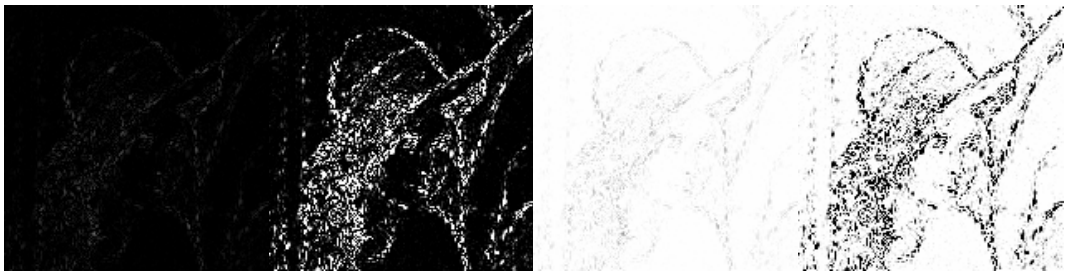
$$S^k(i,j) = \left|\Omega_1(i,j) - \Omega_2(i,j)\right|^k \tag{3.7}$$

$$MSE = \frac{1}{M \times N} \cdot \left(\sum_{i=1}^{M} \sum_{j=1}^{N} \left|\Omega_1(i,j) - \Omega_2(i,j)\right|^2 \right) \tag{3.8}$$

Pro kvalitnější porovnání rekonstrukce jsme si vzorec (3.8) ještě upravili do podoby (3.9), která lépe vystihuje problémy rekonstrukce. Volbou $k = 1$ můžeme zobrazit lineární chybu a volbou $k = 2$ kvadratickou. Analýzou těchto dvou chyb můžeme říct, zda máme víc nebo míň větších chyb neboť s druhou mocninou jsou velké chyby v rozdílu ještě více zvýrazněny.

$$S^k = \frac{1}{T} \cdot \left(\sum_{i=1}^{M} \sum_{j=1}^{N} \left|\Omega_1(i,j) - \Omega_2(i,j)\right|^k \right) \tag{3.9}$$

Dalším parametrem, který jsme upravili je hodnota *T*. Původně byla hodnota $T = M \times N$ což vztahovalo chybu na všechny obrazové body a tedy i ty, kde byl rozdíl jasových hodnot nulový. V našem případě jsem za hodnotu *T* použili počet poškozených pixelů. Vztáhli jsme tak chybu pouze na obrazové body, které byly rekonstruovány.



a) b) c) d)

Obrázek 19: Vizualizace MSE dle (3.7). Chyba se odliší jasem pixelu. (zleva: S, S^2 , 255 - S, 255 - S^2)

Často se provádí vizualizace MSE chyby (3.7). Z obrázků jsou pak patrné oblasti, kde jsou největší problémy a je možné je dále analyzovat. Protože se dále budeme setkávat s hodnotami S^1 a S^2, tak si uvedeme i jejich maximální hodnoty. Maximální hodnota S^1 je 255 a maximální hodnota S^2 je 65025. Těchto mezních hodnot však není možné dosáhnout, neboť by to znamenalo, že mezi originálním obrazovým bodem a rekonstruovaným obrazovým bodem je 255 jasových úrovní (černý pixel byl opraven na bílý). Hodnoty MSE je vhodné občas přepočítat na interval jasových hodnot, neboť hodnoty mohou být větší než maximální hodnota jasu.

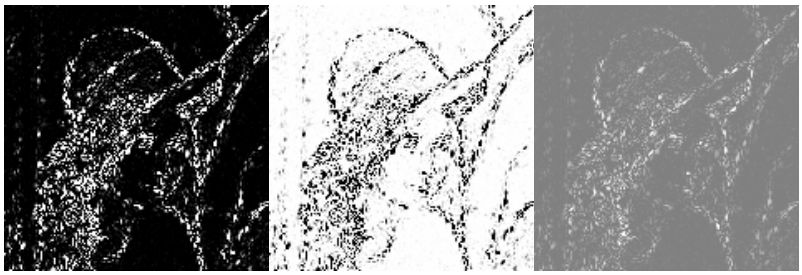
$$PSNR = 20 \log_{10}\left(\frac{255}{RMSE}\right) \tag{3.10}$$

Dalším možným způsobem pro porovnání rekonstruovaného a originálního obrazu je vyjádření PSNR (*peak signal-to-noise ratio*) chyby neboli vrcholový odstup signál-šum (3.10) , které se používá hlavně pro porovnávání obrazů při kódování videa. Nejdříve je

nutné spočítat MSE chybu dle (3.8) a z ní pak RMSE (*root mean square error*), což je druhá odmocnina MSE. Hodnoty PSNR se typicky pohybují v intervalu 20 až 40 dB a jsou udávány na dvě desetinná místa. Samostatná hodnota PSNR není až tak významná. Významné je porovnání dvou hodnot pro různé rekonstruované obrazy. Hranice 50dB se udává jako místo, kde lidské oko přestává vnímat poškození nebo ztrátu informace v obraze. Samozřejmě, že čím je hodnota větší, tím je lepší kvalita rekonstruovaného obrazu.

Jiná důležitá technika pro zobrazení chyby je vytvoření tzv. „chybového obrazu“, který zobrazuje chybu mezi obrazovými body. Nejjednodušší způsob výpočtu tohoto obrazu je výpočet rozdílu mezi originálním a rekonstruovaným obrazem. Takový obraz je však velice nepřehledný (Obrázek 19a) neboť nulová chyba je reprezentována černým pixelem a většina ostatních chyb je malá a reprezentována tak stupni šedi. Typický způsob je pak vynásobení chyby konstantou *k*, která zlepší její viditelnost a posunutí celého obrazu do stupňů šedi (1). Podobnost s obrazem MSE můžete porovnat na níže. (Obrázek 20).

$$\Omega(i,j) = k \cdot [\Omega_1(i,j) - \Omega_2(i,j)] + 128 \qquad \textbf{(1)}$$



Obrázek 20: Porovnání vizualizace MSE chyby (vlevo a uprostřed) a chybového obrazu (vpravo) dle (3.10).

Ještě si uvedeme tabulku označení jednotlivých chybových obrazů, jak je budeme používat dále v textu a v testech.

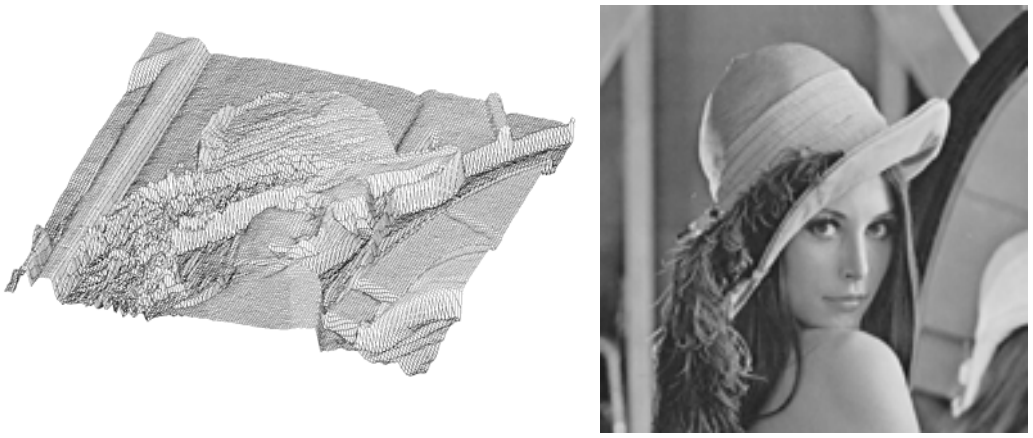
Výpočet	Označení	Ukázka obrazu
S	B	Obrázek 19a
S^2	B2	Obrázek 19b
255-S	W	Obrázek 19c
255-S^2	W2	Obrázek 19d
PSNR	PSNR	Obrázek 20 vpravo

Tabulka 3: Označení chybových obrazů.

V tabulkách shrnujících informace o testování jednotlivých metod budou zpravidla uvedeny údaje S, S^2 a PSNR. U všech těchto hodnot bude taktéž zvýrazněna nejlepší (světle šedá) a nejhorší (tmavě šedá) hodnota pro danou metodu a chybu.

3.6 Rekonstrukce obrazu RBF metodou

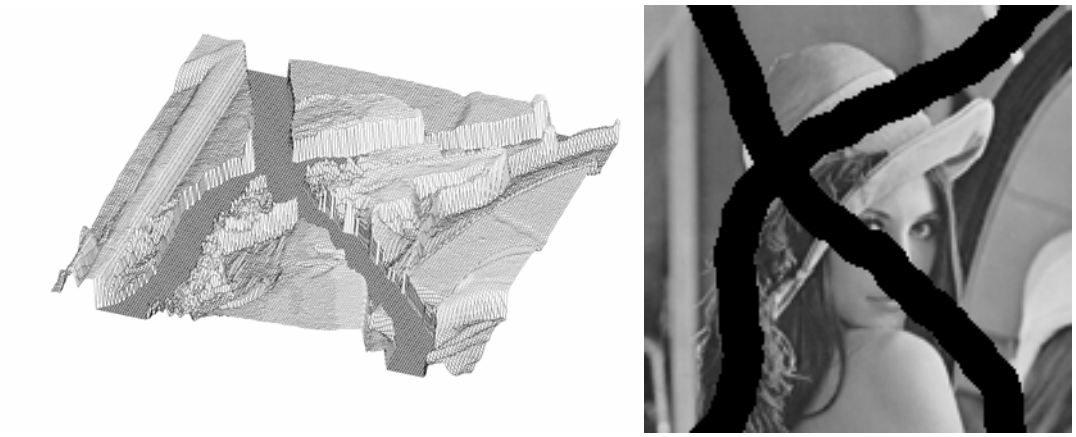
V kapitole 3.3 jsem nadefinovali problém, který chceme řešit RBF metodou a nyní si ukážeme jak bude RBF metoda v tomto případě pracovat. Máme tedy originální obraz (Obrázek 21b), jež si můžeme představit jako povrch (Obrázek 21a), kde hodnota intenzity v jednotlivých obrazových bodech udává výšku ve směru osy *z*.



a) b)

Obrázek 21: Originální obraz - a) jako povrch b) jako obraz.

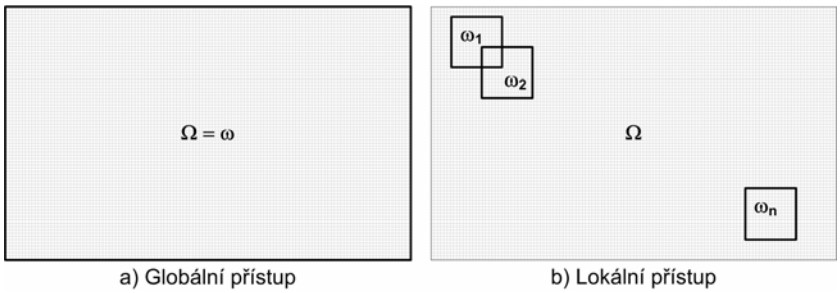
Dále máme masku definující poškozené obrazové body (předpokládáme, že tato maska je získána přímo nebo výpočtem pro daný obraz). Rozdílem masky a originálního obrazu dostaneme poškozený obraz (Obrázek 22). Jelikož známe masku, tak známe i obrazové body, které je potřeba rekonstruovat.



a) b)

Obrázek 22: Poškozený obraz - a) jako povrch b) jako obraz.

Před začátkem rekonstrukce musíme ještě určit přístup, kterou zvolíme pro rekonstrukci. Z hlediska RBF metody existují dva základní přístupy, jak rekonstruovat obraz. Prvním z nich je globální přístup (Obrázek 23a), kdy jsou hodnoty vektoru $\boldsymbol{\lambda}$ spočteny pro všechny známé hodnoty pixelů v celé oblasti Ω. Druhý přístup je lokální, kdy se postupně zpracovávají části obrazu o definované velikosti. Může to být například jen malá submatice (okno) o velikosti například 5x5 pixelů (Obrázek 23b). S touto submaticí se putuje po obraze a dělají se potřebné výpočty. Jestliže máme zvážit výhody a nevýhody obou přístupů, tak hlavním výhodou prvního přístupu je, že potřeba pouze jeden výpočet RBF funkce na obrazu a pak je již možné vyhodnotit funkci v obrazových bodech, kde neznáme hodnotu. Nevýhodou je velká výpočetní náročnost, neboť je nutné řešit systém lineárních rovnic. V druhém případě je velkou výhodou, že je počítán pouze malý lineární systém, ale zato je počítán pokaždé, když se s „oknem“ někam posuneme. Oblast zpracování budeme označovat $\boldsymbol{\omega}_i$, i = 1,...,n, kde n je počet submatic, které jsou při rekonstrukci použity.



Obrázek 23: Různé přístupy k zpracování obrazu.

Oblasti zpracování se mohou překrývat avšak pokaždé musí být sestavena a vypočtena soustava rovnic z hodnot pixelů v nichž je známa hodnota. Zjednodušený algoritmus pro rekonstrukci poškozeného obrazu vypadá takto:

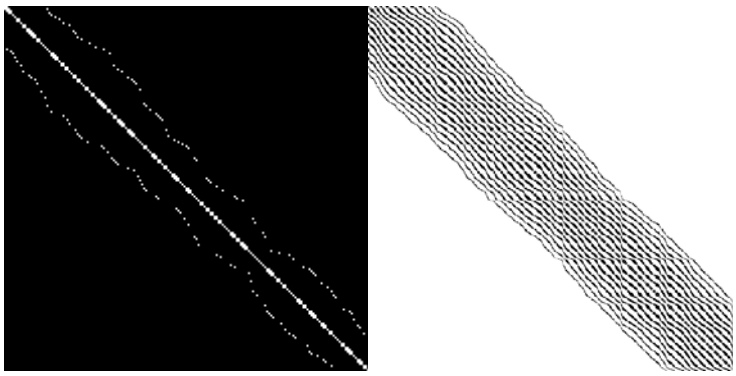
1) definice oblasti zpracování ω
    a. celý obraz
    b. část obrazu – ω_i
2) vytvoření lineárního systému dle (2.13)
3) vypočtení lineárního systému
4) vypočtení hodnot v poškozených obrazových bodech

Algoritmus 1: Zjednodušený návrh algoritmu rekonstrukce obrazu.

Uvedený algoritmus se bude opakovat, pokud je rekonstruována jen část ω_i obrazu dle 1b. Zastavovací podmínkou je informace o počtu rekonstruovaných obrazových bodů. Pokud je počet rekonstruovaných obrazových bodů roven počtu bodů masky, tak může být algoritmus ukončen.

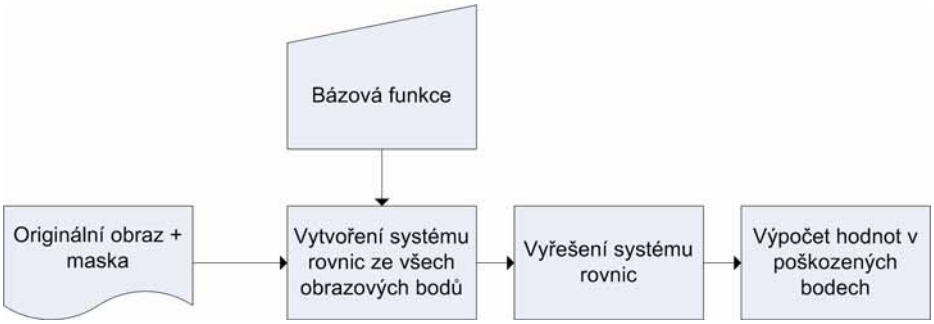
3.6.1 RBF metoda - globální přístup

Jak již bylo uvedeno výše, tak globální metoda má své výhody i nevýhody. Hlavní výhodou je fakt, že celý obraz bude rekonstruován v jednom průchodu. Proti této výhodě je veliká náročnost výpočtu neboť matice **A** (2.12) je „plná“. Jisté urychlení může poskytnout vhodně zvolená bázová funkce ϕ. Neboť například s použitím CSRBF (Tabulka 2) můžeme mít submatice speciální tvar (Obrázek 24).



Obrázek 24: Vyplnění submatice A při použití TPS bázové funkce (vlevo) a *compactly-supported* bázové funkce (vpravo). Nenulové hodnoty jsou tmavé a nulové hodnoty světlé.

Tvar submatice může být využit metodou použitou k řešení lineárního systému rovnic [Laga02]. Uložení takové matice má i velké paměťové nároky. Ke snížení paměťových nároků mohou být opět použity různé speciální metody [Jennings66]. Na následujícím příkladu si ukážeme rekonstrukci poškozeného obrazu globálním přístupem.



Algoritmus 2: Globální rekonstrukce poškozeného obrazu.

Příklad:
Provedeme rekonstrukci poškozeného obrazu globálním přístupem. Bude se jednat o část poškozeného Obrázek 22. Tento obraz má rozlišení 80 x 80 pixelů, což je 6400 obrazových bodů a maska definující poškození obrazu obsahuje 3022 bodů. Znamená to, že v obraze zůstane 3378 korektních bodů, ze kterých bude vytvořen lineární systém (2.13) . Matice **A** bude mít řád 3378 a matice **B** (2.17) bude mít řád 3381, což je 11 431 161 prvků v matici **B**. Pouze na uložení prvků matice B je potřeba 91,5 MB paměti (**double** datový typ). Lineární systém bude řešen například LU faktorizací.

a) b) c) d)

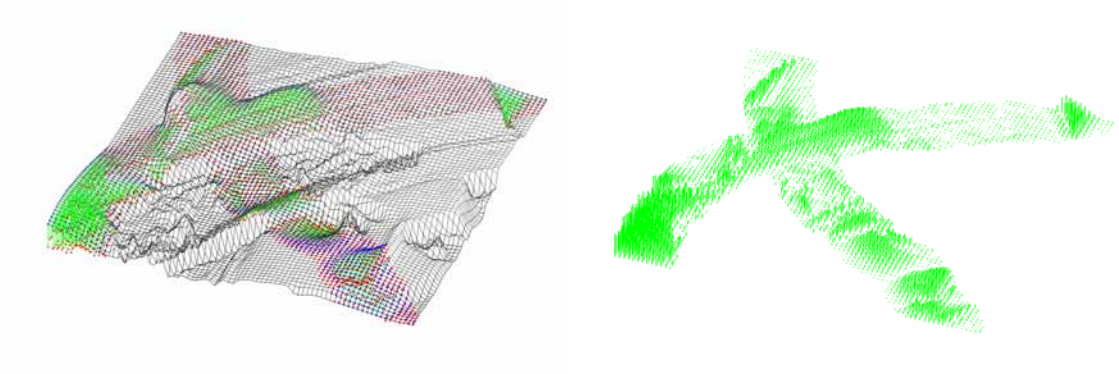
Obrázek 25: Obrázek rekonstruovaný globální metodou – a) originální obraz, b) poškozený obraz, c) rekonstruovaný obraz, d) S^2 obraz

Pro názornost se můžeme ještě podívat na rekonstruovaný obraz jako na 3d povrch (Obrázek 26). Modré body na obrazu vlevo jsou body na povrchu rekonstruovaného obrazu v oblasti masky. Červené body jsou body na povrchu originálního nepoškozeného obrazu. Zelené spojnice mezi nimi ukazují chybu, která je mezi originálním a rekonstruovaným obrazem. Absolutní hodnota chyby je samostatně zobrazena na pravém obrazu.

Následující tabulka shrnuje poznatky získané z rekonstrukce globálním přístupem. Jedná se nejen o vyhodnocení chyb při rekonstrukci, ale také o časy potřebné pro rekonstrukci.

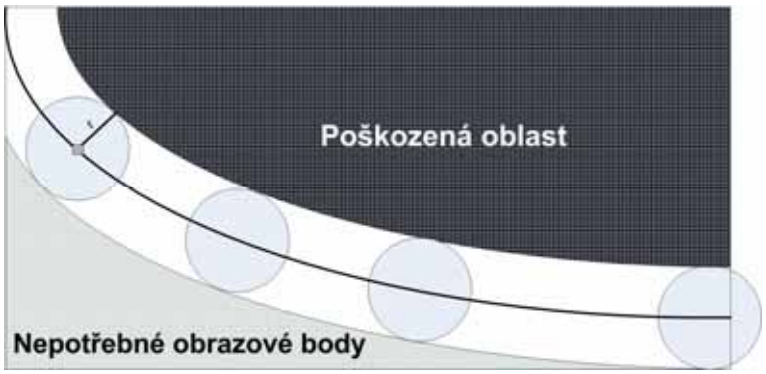
Obraz	S	S^2	var(X)	PSNR	$\phi(r)$	B	Doba výpočtu
Obrázek 25	32,249	2292,528	74,0016	14,5277	TPS	3381	10:40

Tabulka 4: Statistika rekonstrukce globální metodou.



Obrázek 26: Znázornění chyby rekonstrukce.

V případě globálního přístupu s globální funkcí (např. TPS) je nutné použít při rekonstrukci obrazu kompletní obraz, neboť není zaručeno, jaký obrazový bod může ovlivnit výsledek rekonstrukce. V případě použití CSRBF je možné definovat okolí poškozené části, které je třeba dodržet (Obrázek 27). Písmeno **r** značí *radius-of-support* CSRBF funkce.



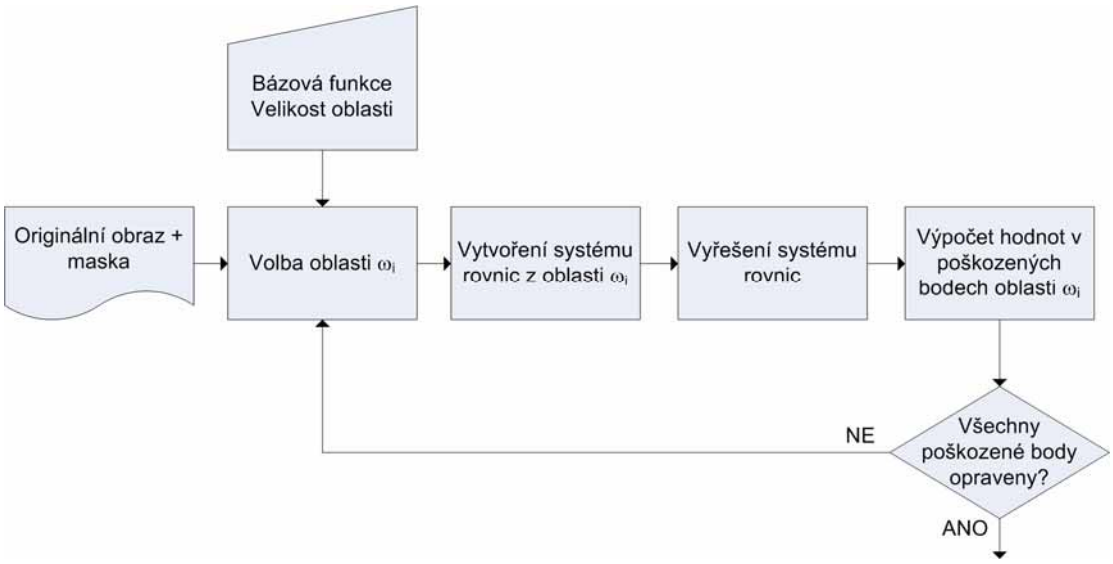
Obrázek 27: Definice okolí poškozené části, které je nutné dodržet pro korektní rekonstrukci u *compactly-supported* bázových funkcí.

Z předchozího příkladu je vidět, že globální přístup není obecně použitelný na rekonstrukci poškozených obrazů. Globální přístup brzy naráží na paměťové a výpočetní nároky RBF metody. Snad pouze při použití CSRBF metody, speciálních schémat na ukládání řídké matice a výpočtových metod, které dokáží řídkost matice využit, by byl globální přístup použitelný pro větší obrazy.

3.6.2 RBF metoda - lokální přístup

Lokální přístup na rozdíl od globálního opravuje poškozené obrazové body postupně. Základní princip spočívá v „putujícím okně“ o určité velikosti. Výpočetní náročnost není rozhodně tak veliká jako v případě globální metody avšak základní část rekonstrukce se musí opakovat pro každou oblast ω_i, i=1,…,n.

S tímto přístupem se objevuje několik parametrů, které je nutné nastavovat a sledovat neboť ovlivňují kvalitu výsledné rekonstrukce. Jsou to parametry jako velikost okna, způsob pohybu po obraze a způsob opravy poškozených obrazových bodů.



Algoritmus 3: Lokální rekonstrukce poškozeného obrazu.

Algoritmus 3 ukazuje postup výpočtu při použití lokálního způsobu rekonstrukce. Můžeme vidět, že vstupem je stejně jako v případě globálního přístupu originální obraz a maska poškození. Dalším a novým vstupním parametrem je velikost oblasti, se kterou chceme pracovat. Jedná se o velikost submatice ω_i. Základní opakující se cyklus obsahuje kroky jako vybrání oblasti s celkové oblasti zpracování Ω (Obrázek 17), vytvoření systému rovnic pro oblast ω_i, vyřešení systému lineárních rovnic pro zvolenou oblast, vypočtení hodnot poškozených obrazových bodů v oblasti ω_i a nakonec je rozhodovací pravidlo, které v případě, že nejsou rekonstruovány všechny poškozené obrazové body v oblasti Ω, rozhodne o opakování cyklu od výběru další oblasti ω_{i+1} obsahující poškozené obrazové body. Celý cyklus se bude opakovat dokud nebude provedena kompletní rekonstrukce.

Na rozdíl od globálního přístupu můžeme v lokálním přístupu definovat okolí poškozené části obrazu, jež má ještě smysl brát v úvahu, prakticky pro libovolnou bázovou funkci. Je to dáno tím, že rekonstrukce se provádí pro každou oblast zvlášť, tudíž obrazové body, které jsou mimo oblast ω_i nemají žádný vliv na rekonstrukci v oblasti.

Všechny uvedené parametry lokálního přístupu budeme podrobněji zkoumat v dalších kapitolách, kde se budeme ještě zabývat i vlivem volby bázové funkce na aproximaci jasových hodnot obrazových bodů.

3.6.3 Analýza rekonstrukce

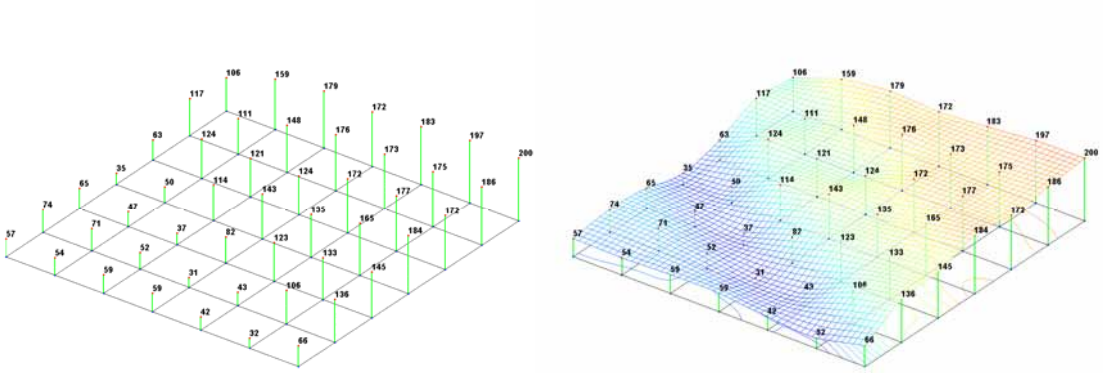
V této kapitole se budeme zabývat základními parametry lokálního přístupu k rekonstrukci poškozených obrazů. Přes to, že parametry jsou shodné pro oba uvedené přístupy, tak neshledáváme globální přístup kvůli jeho náročnosti až tak zajímavým pro tuto oblast. Stejně jak pro oblast rekonstrukce poškozených obrazů se globální přístup stal nezajímavým i pro rekonstrukce a implicizaci 3d objektů. V oblasti 3d vizualizace a rekonstrukce se postupně došlo taktéž na rozdělení problému na menší části a jejich následnému samostatné zpracování.

3.6.3.1 Volba bázová funkce

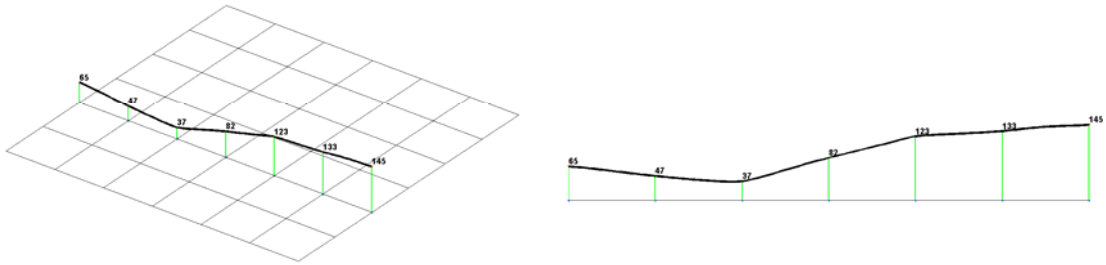
Volba bázové funkce je velmi důležitý aspekt ovlivňující kvalitu rekonstrukce. Jak bylo již zmíněno výše, tak prakticky jediná práce [Savchenko02] zabývající se rekonstrukcí poškozených obrazů RBF metodou využívá CS bázové funkce k rekonstrukci. Dokážeme,

že volba CSRBF není nejlepší neboť CSRBF nedokáží dobře rekonstruovat určité druhy poškození (Obrázek 14).

Nejprve by bylo dobré vidět, jak se chovají různé bázové funkce (Tabulka 1 a Tabulka 2, kapitola 2.5) při interpolaci dat neboli jasových hodnot obrazu. Obrázek 28 vlevo ukazuje malou oblast obrazu o velikosti 7x7 obrazových bodů i s hodnotami jasu v jednotlivých bodech. Interpolací jasových hodnot RBF metodou a následným dopočtením hodnot mezi známými jasovými hodnotami v jednotlivých bodech dostaneme plochu znázorněnou na obrázku 28 vpravo.



Obrázek 28: Definice části obrazu s jasovými hodnotami (vlevo) a jejich interpolace RBF metodou (vpravo).



Obrázek 29: Řez interpolovanými jasovými hodnotami z Obrázek 28.

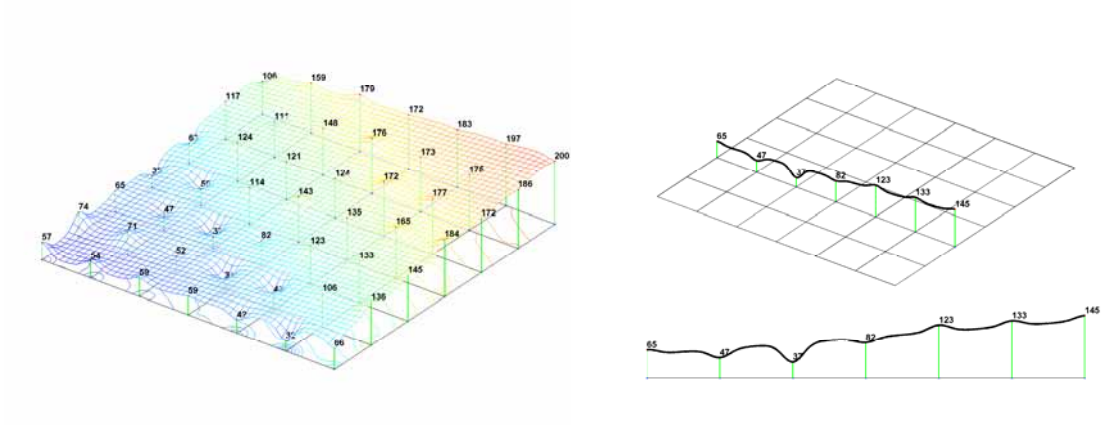
Takto vypadá interpolace pokud máme plný obraz, tzn. všech 49 jasových hodnot v bodech. V případě rekonstrukce však neznáme jasové hodnoty některých obrazových bodů a snažíme se je dopočítat ze znalosti jejich okolí. Pokud bychom tedy měli poškozený prostřední obrazový bod na obrázku 28, tak bychom měli k dispozici 48 hodnot ze 49 a neznámou hodnotu bychom dopočítávali ze 48-okolí poškozeného bodu. Uvedený příklad nám bude sloužit pro vizuální porovnání kvality interpolace a na malém výřezu o rozměrech 7x7 obrazových bodů budou testovány i ostatní bázové funkce.

Jestliže budeme uvažovat o volbě bázové funkce, tam musíme vzít v úvahu několik základních pravidel, která by bázová funkce měla splňovat.

1) kvalitní výsledky při rekonstrukci
2) nezávislost na velikosti zvoleného n-okolí
3) nezávislost na vzdálenosti mezi známými hodnotami
4) nezávislost kvality rekonstrukce na volbě parametru funkce

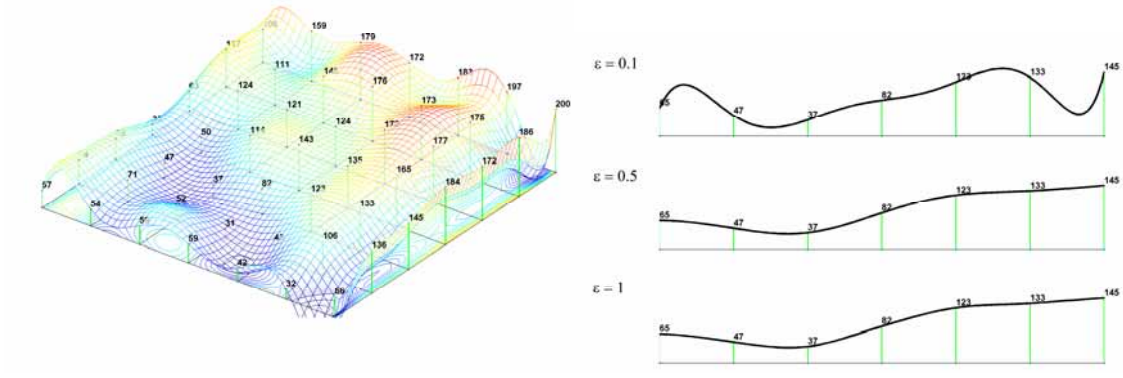
Uvedeme si stručný přehled bázových funkcí (podrobný popis lze nalézt v kapitole 2.5) i s ukázkami interpolace na výřezu o velikosti 7x7 obrazových bodů.

První ukázka interpolace se bude týkat GRBF. V dostupné literatuře zabývající se bázovými funkcemi Gaussova typu [Schagen79, Platte05] není dostatečně zodpovězena otázka volby volného parametru ε (Tabulka 1). Můžeme sice experimentálně stanovit optimální hodnotu parametru, avšak tato hodnota se bude měnit pro různá vstupní data [Schagen79]. Platte R.B. a Driscoll T.A. popsali některé vlastnosti interpolace pomocí GRBF a určili podmínky stabilní aproximace při určitém rozložení bodů [Platte05]. Přesto může dojít singularitě lineárního systému. Stanovení parametru může být potencionálním směrem dalšího výzkumu.



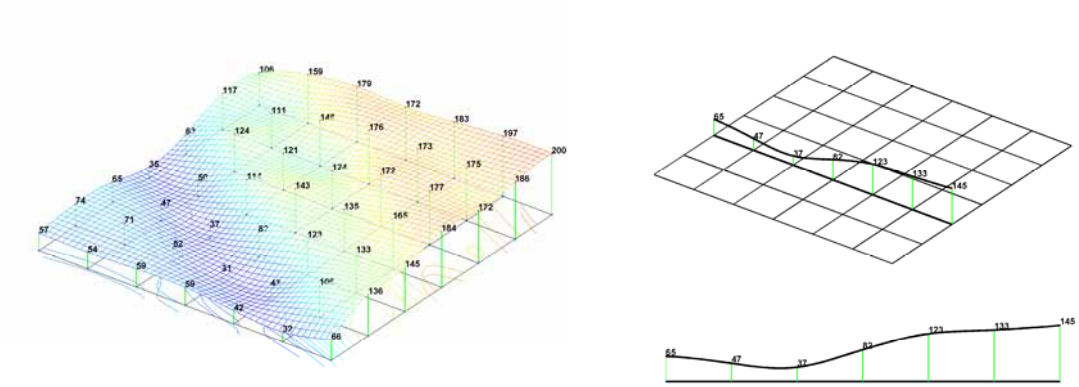
Obrázek 30: Interpolace dat GRBF s nevhodným parametrem ε (ε = 5).

Parametr ε se nachází taktéž v definici IQ, IMQ a MQ (Obrázek 31) bázových funkcí. Stejně jak je tomu u bázových funkcí Gaussova typu tak i zde je použití parametru závislé na datech a stejné doporučení o experimentálním stanovení parametru udává i Hardy [Hardy90]. Fornberg udává, že změnou parametru ε se pohybuje řešení u MQ bázové funkce mezi přesností pro malé ε a dobrou podmíněností řešení pro velké ε [Fornberg02, Figure 4.1(c)]. Stabilitou výpočtu při použití MQ bázové funkce se zabývá [Fornberg04]. Podmíněnost řešení pro IQ a IMQ bázová funkce klesá ještě rychleji pro $\varepsilon \to 0$ než u MQ bázové funkce.



Obrázek 31: Interpolace dat MQ bázovou funkci s různými hodnotami parametru ε. (vlevo ε = 0.1)

Další možnou volbou bázové funkce jsou lineární, kvadratické, kubická a nebo také bázová funkce pátého řádu $\phi(r) = r^5$ (*quantic*). O těchto bázových funkcích a jejich použití se literatura příliš nezmiňuje. Pouze [Fornberg02] se zmiňuje o jejich spojitosti s přirozeným spline.

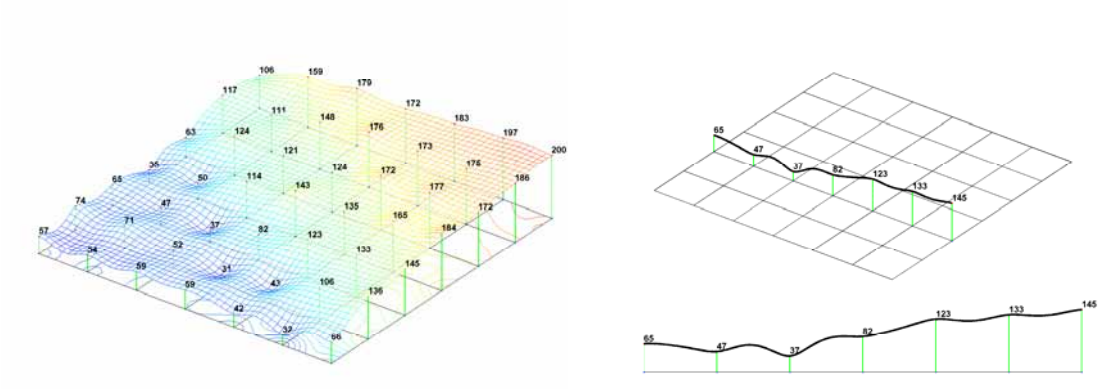


Obrázek 32: Interpolace dat *quantic* bázovou funkci.

Zajímavou bázovou funkcí je TPS. Je velmi často používána a má fyzikální opodstatnění [Truk99]. Poskytuje velice kvalitní výsledky a nemá žádný volitelný parametr. Lineární systém je plný při jejím použití, proto ze většinou nepoužívá při globálním přístupu na rozdíl od CSRBF.

Poslední skupinou jsou CSRBF bázové funkce. Tyto bázové funkce jsou velmi populární hlavně kvůli řídkosti lineárního systému (2.12) a s jejich použití se stala reálnou možnost interpolovat obrovské množství roztroušených bodů [Tobor04, Othake04a, Othake04b]. Existují různé urychlovací techniky využívající jejich lokalitu. Jistou nevýhodou je nutnost použití parametru α, kterým se funkce tvaruje. Tento parametr se však dá nastavit automaticky pro danou bázovou funkci.

Zde bychom se měli zmínit, že Savchenko V. i Kojekine N. používají CSRBF na rekonstrukci poškozených obrazových bodů v globálním přístupu [Savchenko02, Kojekine04]. Standardně mají obraz v intervalu <0,1> a například pro obraz o velikosti 300x300 obrazových bodů používají *radius of support* o velikosti 0.03. To znamená, že vliv bázové funkce je do vzdálenosti 3% a tedy 9 obrazových bodů.

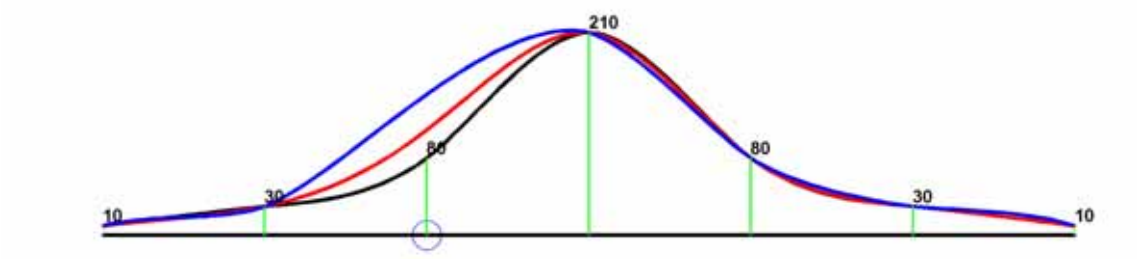


Obrázek 33: Interpolace CSRBF s nevhodně nastaveným poloměrem parametrem α.

Z uvedených bázových funkcí jsme vyřadili bázové funkce Gaussova typu, MQ, IMQ a IQ. Gaussovské exponenciální funkce, které jsou nevhodné z hlediska stability výpočtu a také nutností vhodné volby parametru ε. Ze stejného hlediska musíme vyřadit i MQ, IMQ a IQ bázové funkce. Samozřejmě, že by bylo možné systematickým způsobem měnit parametr ε a určit tak nejvhodnější interpolací s danou funkcí avšak tento proces je zdlouhavý a úspěch není zaručen. A to nejen v přesnosti řešení, ale také v jeho stabilitě. Dále vyřadíme

lineární v jejímž případě by se jednalo pouze o klasickou lineární interpolaci, kvadratickou bázovou funkci, která je nevhodná pro interpolaci dvourozměrných dat. Zůstali nám tedy TPS, CSRBF, kubická a *quantic* bázová funkce. Těmito bázovými funkcemi se budeme zabývat dále. Jednoduchou ukázku interpolace různými bázovými funkcemi naleznete v Příloze B.

V Příloze B je uvedeno, jak se některé bázové funkce chovají na intervalu mezi zadanými hodnotami a to někdy velmi zajímavě. Podobným způsobem se budou chovat i v případě, že nebudou známy některé hodnoty a bude nutné je dopočítat z interpolace (Obrázek 34). Podle této vlastnosti omezíme bázové funkce, o které se budeme zajímat.

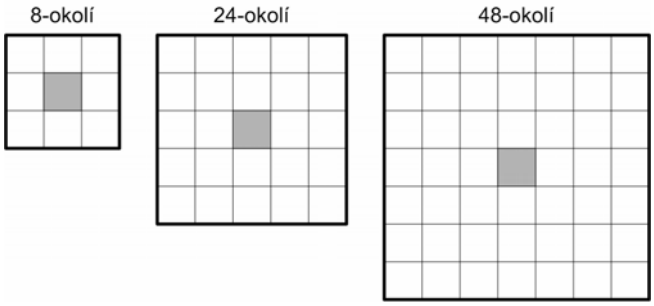


Obrázek 34: Různé interpolace dat TPS bázovou funkcí: černá – všechny hodnoty, červená – bez zakroužkované hodnoty se znalostí okolí, modra – bez zakroužkované hodnoty pouze v řezu.

Důležitým faktorem pro kvalitu interpolace je i znalost okolí. Jak si můžeme všimnout na obrázku uvedeném výše, tak v případě znalosti okolí (červená křivka) je odlišnost interpolace bez zakroužkované hodnoty od originální menší než v případě, že bychom prováděli interpolaci pouze v řezu (modrá křivka). Podrobnou ukázku interpolace chybějících hodnot naleznete v Příloze C.

3.6.3.2 Volba okolí pro rekonstrukci

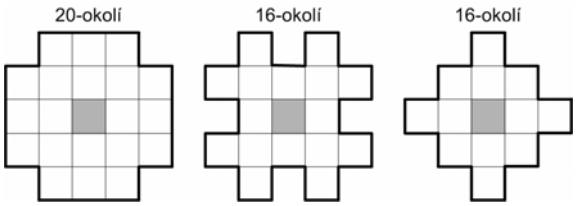
Dalším důležitým faktem je volba velikosti okna. Jak bylo zmíněno již dříve, tak lokální metoda je založena na posouvajícím se okně o určité velikosti. V tomto okně je pak prováděna interpolace poškozených obrazových bodů pouze ze známých hodnot bodů v tomto okně. Na velikosti okna a i jeho tvaru závisí kvalita výsledné interpolace. Různé tvary a velikosti se hodí pro různé druhy obrázků a různé způsoby použití RBF metody v oblasti zpracování obrazu.



Obrázek 35: Klasické čtvercové okolí zpracovávaného bodu.

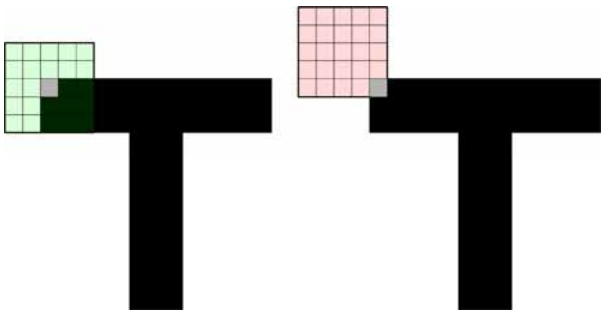
Jelikož se provádí interpolace podle informace v prostředním bodě okna, tak nejmenší velikost čtvercového okolí je 8-okolí (Obrázek 35a). Bylo vyzkoušeno mnoho různých

vzorů (Obrázek 35, Obrázek 36), ale všechny provedené testy na kvalitu výsledné interpolace byly nejlepší pro klasické čtvercové 24-okolí (Obrázek 35b).



Obrázek 36: Různá alternativní okolí zpracovávaného bodu.

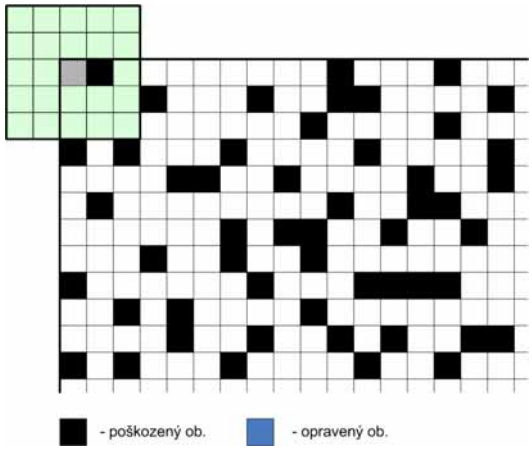
Jak bylo zmíněno výše, tak je vždy rekonstruován střed okna. Může se zdát, že tato volba není vhodná pro některé konfigurace bodů (Obrázek 37). Výpočet by mohl být totiž proveden z mnohem většího počtu známých bodů, avšak tato volba má hlubší význam. Jak udává dostupná literatura, tak většina bázových funkcí se chová nepředvídatelně na okrajích oblasti interpolace [Fornberg92]. Proto by se mohlo stát, že na okrajích bude docházet k větší chybě.



Obrázek 37: Volba způsobu rekonstrukce poškozeného obrazového bodu.
a) námi zvolený způsob ve středu oblasti interpolace – 16 známých hodnot,
b) možný způsob se znalostí většího počtu hodnot – 24 známých hodnot.

3.7 Navržené metody pro rekonstrukci poškozených obrazů

V následující části se budeme zabývat metodami rekonstrukce, které využívají výše popsanou RBF metodu. Bylo navrženo několik základních metod k rekonstrukci poškozených obrazových bodů při lokálním přístupu.

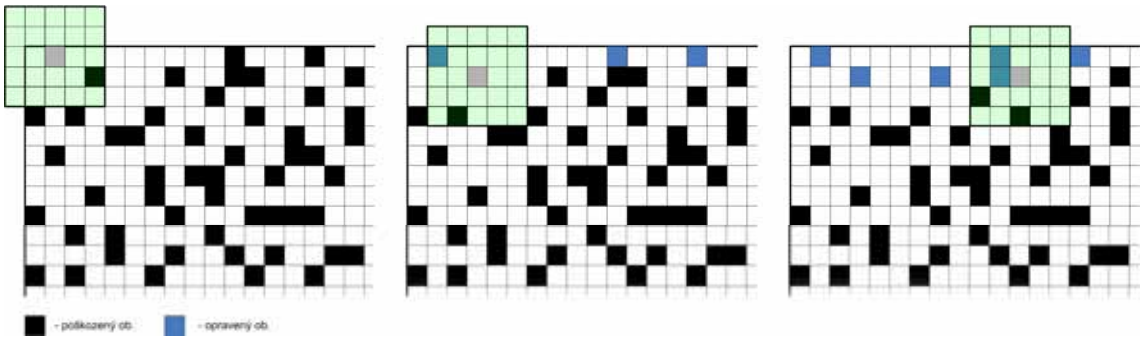


Obrázek 38: Základní konfigurace okna na začátku rekonstrukce.

Obrázek 38 ukazuje standardní začátek rekonstrukce poškozeného obrazu při použití 24-okolí a scan-line metody. Z tohoto obrázku budeme vycházet a ukážeme si na něm, jak jednotlivé metody pracují. Ve většině tabulek jsou zvýrazněny nejlepší (světle šedá) a nejhorší (tmavě šedá) hodnoty.

3.7.1 Scan-line metoda – přímá rekonstrukce

Tato metoda je založena na přímé rekonstrukci poškozených obrazových bodů v jednom průchodu obrazem. Nově vypočtené hodnoty poškozených ob. jsou po vypočtení okamžitě doplňovány do obrazu a hned v dalším kroku je možné s nimi počítat (Obrázek 39). U popisu obrázku je uvedeno z kolika hodnot v n-okolí je počítána interpolace.



a) **b)** **c)**

Obrázek 39: Kroky metody přímé rekonstrukce: a) rekonstrukce prvního poškozeného bodu – 23 známých hodnot, b) zde je již použita hodnota z předcházejícího kroku – 23 známých hodnot, c) další postup metody – 22 známých hodnot.

Může být samozřejmě namítnuto, že při výpočtu vznikne chyba, která je posléze rozšířena do dalšího výpočtu, přesto tato metoda poskytuje poměrně kvalitní a hlavně rychlý výsledek v jednom průchodu. Algoritmus celého výpočtu vypadá následovně:

```
Vstup: O - poškozený obraz, M - maska, k - n-okolí, ϕ - bázová funkce
Výstup: rekonstruovaný obraz

for i:1:M
 for j:1:N /*přes všechny obrazové body*/
  if(M(i,j) == díra)    /*na dané pozici je díra*/
   v = OkoliBodu(i,j,O,k);
   B = VytvorMatici(v,ϕ);
   b = VektorPravychStran(v);
   lambda = ResSystem(B,b);
   O(i,j) = HodnotaBodu(i,j,lambda,v,ϕ);
   ZrusDiru(M,i,j);
  end;
 end;
end;
```

Algoritmus 4: Scan-line metoda přímé rekonstrukce.

Paměťová náročnost metody je $O(kn)$, kde $n = M \times N$ a k je koeficient zahrnující vliv masky, maticový systém a vektor okolí. Algoritmická složitost má trochu složitější zápis a je $O(knl^4)$. Pro každý bod obrazu, který považujeme za díru, vypočteme lineární systém z l známých hodnot v jeho okolí (maximálně 24 hodnot pro 24-okolí). Výpočet lineárního systému LU faktorizací má algoritmickou složitost $O(n^3)$ a proto je ve vyjádření algoritmické složitosti našeho algoritmu l^4. Koeficient k definuje míru poškození obrazu a platí, že $k < 1$.

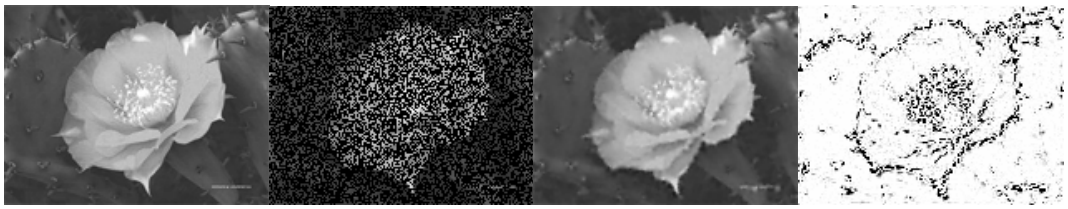
Testování metody

Obraz	S	S^2	PSNR	B	ϕ(r)	Čas [s]
1	6,455	130,766	26,966	15360	1	2,063
2	8,173	173,050	25,749	4884	1	0,781
3	19,036	929,714	18,447	6477	1	0,625
4	40,541	3325,686	12,912	2978	1	0,266
5	6,690	195,388	25,222	11520	1	1,500
6	7,554	210,206	24,904	3784	1	0,594
7	16,519	952,787	18,341	5309	1	0,516
8	5,281	90,595	28,560	11520	1	1,469
9	6,465	107,881	27,801	3784	1	0,594
10	14,672	576,893	20,520	5309	1	0,516
11	47,011	7596,265	9,325	11520	1	1,469
12	49,540	8201,704	8,992	3784	1	0,594
13	65,684	13042,939	6,977	5309	1	0,516
14	17,467	540,600	20,802	17161	1	2,047
15	10,646	250,660	24,140	212880	1	27,750

Tabulka 5: Testování metody přímé rekonstrukce.

Obrázek 40: Ukázka rekonstrukce přímou metodou obrazu 1. (zleva: originální obraz, poškozený obraz, rekonstruovaný obraz, B2 obraz)

Tato nejjednodušší metoda poskytuje poměrně kvalitní výsledky za rozumnou dobu výpočtu. Doba výpočtu se pohybuje v intervalu 0,266 s až 27,875. Přičemž nejdelší doba výpočtu je u obrazu Marsu, který má celkem 354760 obrazových bodů, které je nutné projít, aby bylo možné provést rekonstrukci. Musíme také uvažovat, že takovéto poškození obrazu je opravdu extrémní a může se vyskytovat například při poškození CCD prvku digitálního fotoaparátu. Nejlépe zrekonstruovaný obraz je obraz č.8 a můžete si ho prohlédnout na Obrázek 41.

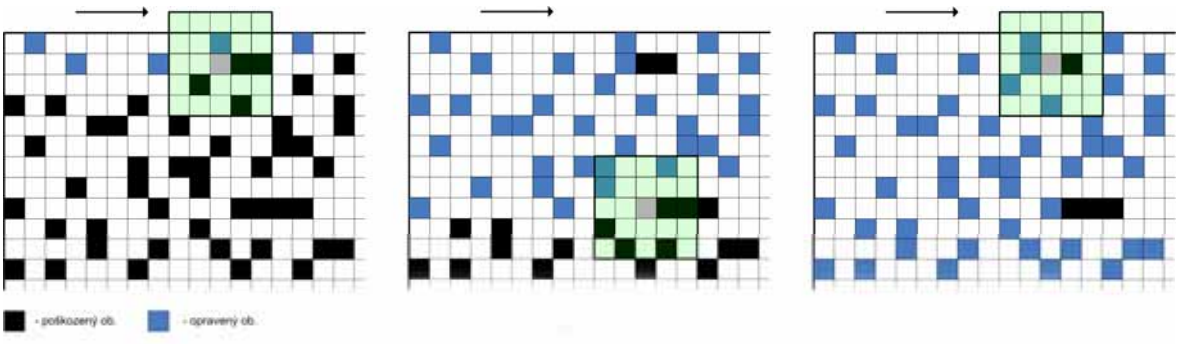


Obrázek 41: Nejlépe zrekonstruovaný obraz z Tabulky 4 přímou metodou. (zleva: originální obraz, poškozený obraz, rekonstruovaný obraz, B2 obraz)

Pokud jde vše bez problémů, tak uvedená metoda prochází poškozený obraz pouze jednou. Může se však stát, že ve zpracovávané oblasti je nedostatečné množství známých hodnot a lineární systém je neřešitelný. Pokud není možné vypočítat hodnotu obrazového bodu kvůli neřešitelnosti sytému, tak je proveden další průchod obrazem a neznámá hodnota je spočtena v dalším průchodu.

3.7.2 Scan-line metoda – rekonstrukce zleva, zprava

Tato metoda je založena na postupné rekonstrukci poškozeného obrazu z jedné strany obrazu. Oba případy zleva i zprava jsou prakticky identické a proto se budeme dále zmiňovat již pouze o rekonstrukci z jedné strany obrazu (zleva). Základním pravidlem pro tuto metodu je zastavení rekonstrukce, pokud jsou následující dva obrazové body díry tzn., že musí za sebou následovat větší množství poškozených obrazových bodů. Pro 24-ti okolí jsou to minimálně 3 obrazové body. Bod určený k rekonstrukci plus další dva ve směru rekonstrukce. Obrazový bod, na kterém došlo k zastavení rekonstrukce a posunutí na další řádek je rekonstruován avšak další body v řádku jsou rekonstruovány až v dalším průchodu. Z uvedeného popisu tedy vyplývá, že metoda je víceprůchodová viz Obrázek 42.



a) b) c)

Obrázek 42: Kroky metody postupné rekonstrukce zleva: a) pokud je za sebou více poškozených ob., tak algoritmus pokračuje na dalším řádku, b) další větší poškozená část obrazu, c) pokračování rekonstrukce v dalším průchodu.

Zdá se, že tato metoda nepřináší žádné zlepšení, ale opak je pravdou. Pokud se totiž podíváte na Obrázek 42a, tak hodnota následujícího obrazového bodu by byla vypočtena z 16-ti známých hodnot ve 24-okolí. Pokud však uplatníme pravidlo o posunu na další řádek v případě velkého množství poškozených obrazových bodů ve směru rekonstrukce a budeme hodnotu následujícího obrazového bodu počítat až v další iteraci (Obrázek 42c), tak bude jeho hodnota počítána z 18-ti známých hodnot. Jedná se vlastně o šíření dopočtených hodnot z oblastí s menším poškozením obrazu. Na ukázce kroků metody je vidět jak jsou části obrazu s menším poškozením rekonstruovány rychleji.

Samozřejmě, že může existovat několik dalších modifikací nebo kombinací průchodu přes obrázek z různých stran a v různém pořadí.

```
Vstup: O - poškozený obraz, M - maska, k - n-okolí, ϕ - bázová funkce
Výstup: rekonstruovaný obraz

do
 for i:1:M
  for j:1:N /*přes všechny obrazové body*/
   if(M(i,j) == díra)    /*na dané pozici je díra*/
    v = OkoliBodu(i,j,O,k);
    B = VytvorMatici(v,ϕ);
    b = VektorPravychStran(v);
    lambda = ResSystem(B,b);
```

```
      O(i,j) = HodnotaBodu(i,j,lambda,v,ϕ);
      ZrusDíru(M,i,j);
      if(KontrolaPoctuDerVpravo(i,j,O,M,k)) break;
     end;
    end;
   end;
  while(!JsouJesteNejakeDiry(M))
```

Algoritmus 5: Scan-line metoda rekonstrukce zleva.

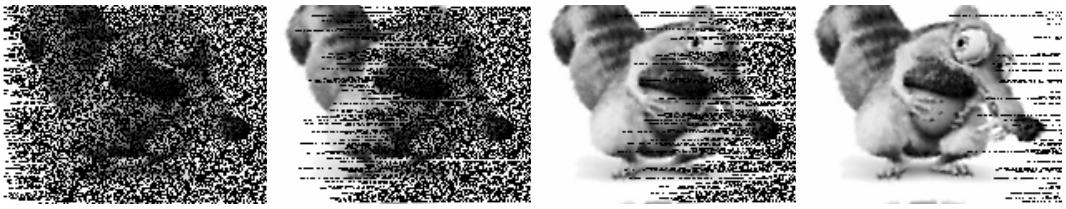
Algoritmus přímé rekonstrukce byl rozšířen o kontrolu podmínky uvedené výše. Tato podmínka je kontrolována funkcí `KontrolaPoctuDer()` a vyvolá posunutí výpočtu na další řádek. Výpočet je prováděn dokud jsou v masce ještě díry.

Paměťová náročnost metody je shodná s metodou přímé rekonstrukce. Algoritmická složitost bude o něco horší neboť v případě velkého poškození může dojít k tomu, že počet iteraci bude jen o málo menší než počet obrazových bodů obrazu ve směru rekonstrukce. Algoritmická složitost je $O(k(hnl^4))$, kde n je hlavní výpočet přes celý obraz a l^4 v sobě zahrnuje vytvoření lineárního systému z l hodnot vektoru $\mathbf{v}$ (maximálně 24 hodnot pro 24-okolí) a výpočet lineárního systému například LU faktorizací. Koeficient k definuje míru poškození obrazu a platí, že $k < 1$. Koeficient h zahrnuje počet iterací výpočtu.

Testování metody

Obraz	S	S^2	PSNR	B	ϕ(r)	Čas [s]	Iterací
1	6,437	129,948	26,993	15360	1	2,172	67
2	8,148	172,640	25,759	4884	1	0,844	41
3	21,100	1161,591	17,480	6477	1	0,734	85
4	45,173	3835,641	12,292	2978	1	0,328	49
5	6,616	191,083	25,319	11520	1	1,656	71
6	7,506	209,647	24,916	3784	1	0,750	41
7	20,752	1568,646	16,176	5309	1	0,594	85
8	5,296	91,691	28,508	11520	1	1,641	71
9	6,489	108,811	27,764	3784	1	0,656	41
10	15,450	624,572	20,175	5309	1	0,594	85
11	47,003	7599,810	9,323	11520	1	1,672	71
12	49,508	8216,054	8,984	3784	1	0,641	41
13	57,965	11848,340	7,394	5309	1	0,609	85
14	18,153	586,088	20,451	17161	1	2,484	75
15	10,622	249,220	24,165	212880	1	30,813	249

Tabulka 6: Testování metody rekonstrukce zleva.



Obrázek 43: Ukázka postupné rekonstrukce obrazu zleva/zprava. (Obraz č.5)

Pokud si uvědomíme, že obraz nemá většinou čtvercový tvar, tak je zřejmé, že budeme-li provádět rekonstrukci ve směru širší strany, tak bude nutný větší počet iterací. Tuto závislost jsme zaznamenali v následující tabulce.

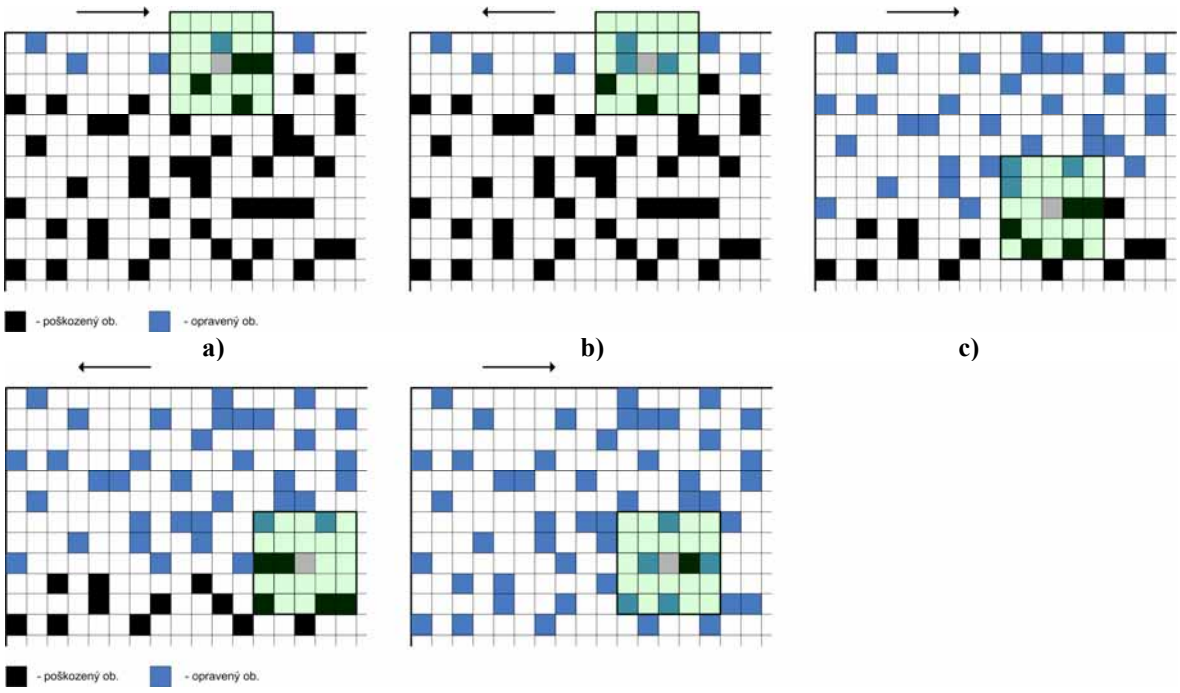
Obraz	S	S^2	PSNR	B	$\phi(r)$	Čas [s]	Iterací
15	10,622	249,220	24,165	212880	1	30,813	249
15 otočený o 90°	10,659	251,425	24,127	212880	1	30,547	139

Tabulka 7: Změna strany rekonstrukce.

Z uvedeného je zřejmé, že z hlediska počtu iterací je výhodné provádět rekonstrukci obrazu ve směru kratší strany obrazu. Ostatní hodnoty výpočtu jsou srovnatelné u obou výpočtů.

3.7.3 Scan-line metoda – vícestranná

Metoda vícestranné rekonstrukce kombinuje rekonstrukce postupující z více stran obrazu. Z hlediska obrazu tedy mohou být rekonstrukce zprava, zleva, shora a zdola. Je zde zřejmá závislost mezi počtem iterací a počtem použitých stranových rekonstrukcí. Metoda je rychlejší než jednostranná metoda a poskytuje o něco lepší výsledky. Jako základ pro vysvětlení jsem si vybrali kombinaci rekonstrukce zprava a zleva, ale budeme testovat i jiné kombinace stranových rekonstrukcí.



Obrázek 44: Kroky metody postupné rekonstrukce zleva a zprava: a) pokud je za sebou více poškozených ob., tak algoritmus zleva pokračuje na dalším řádku, b) algoritmus zprava však nemá v této oblasti problémy, c) a d) problematické místo pro algoritmus zleva i zprava, e) v dalším průchodu je problematické místo již zcela opraveno.

Výhodu oboustranné metody můžeme pozorovat u obrázků s větší mírou poškození v jedné části obrazu jako je například Obrázek 45. V takovém případě je do poškozené oblasti šířena informace z obou stran. Orientací obrazu můžeme ovlivnit kvalitu rekonstrukce obrazu neboť velké poškození orientované ve směru postupu metody není příliš vhodné pro tento druh metody. Nedá se však jednoznačně zamítnout neboť je nutné brát v úvahu i okolí poškození. Pokud bude metoda šířit do oblasti tmavé světlé hodnoty, tak bude chyba rekonstrukce růst (Obrázek 45, Tabulka 8). Samozřejmě že pokud neznáme originální obraz, tak nevíme jaká intenzita byla v oblasti poškození a nemůžeme jednoznačně rozhodnout z jakého směru rekonstruovat.

Obrázek 45: Rekonstrukce poškozeného obrazu (vlevo) metodou zprava-zleva (uprostřed) a metodou shora-zdola (vpravo).

Obrázek 45	S	S^2	PSNR	B	$\phi(r)$	Čas [s]	Iterací
zprava-zleva	31,803	1969,825	15,187	3481	1	0,406	62
shora-zdola	38,549	2634,118	13,924	3481	1	0,328	15

Tabulka 8: Vyhodnocení rekonstrukce pro Obrázek 45.

```
Vstup: O - poškozený obraz, M - maska, k - n-okolí, ϕ - bázová funkce
Výstup: rekonstruovaný obraz

do
 for j=0:1:j<N
  for i=0:1:i<M /*po řádku zleva*/
   if(M(i,j) == díra)    /*na dané pozici je díra*/
    v = OkolíBodu(i,j,O,k);
    B = VytvořMatici(v,ϕ);
    b = VektorPravychStran(v);
    lambda = RešSystém(B,b);
    O(i,j) = HodnotaBodu(i,j,lambda,v,ϕ);
    ZrušDíru(M,i,j);
    if(KontrolaPočtuDěrVpravo(i,j,O,M,k)) break;
   end;
  end;

  for i=M-1:1:0 /*po řádku zprava*/
   if(M(i,j) == díra)    /*na dané pozici je díra*/
    v = OkolíBodu(i,j,O,k);
    B = VytvořMatici(v,ϕ);
    b = VektorPravychStran(v);
    lambda = RešSystém(B,b);
    O(i,j) = HodnotaBodu(i,j,lambda,v,ϕ);
    ZrušDíru(M,i,j);
    if(KontrolaPočtuDěrVlevo(i,j,O,M,k)) break;
   end;
  end;
 end;
while(!JsouJestěNějakéDíry(M))
```

Algoritmus 6: Scan-line metoda rekonstrukce z různých stran obrazu (zleva-zprava).

Paměťová náročnost algoritmu je shodná s předchozími metodami. Algoritmická složitost metody závisí na počtu stran, ze kterých je prováděna rekonstrukce. Výraz algoritmické složitosti je tedy

$$O\left(k\left(\left(\frac{h}{m}\right)nl^4\right)\right), \qquad \textbf{(2)}$$

kde n je hlavní výpočet přes celý obraz ($n = M \times N$) a l^4 v sobě zahrnuje vytvoření lineárního systému z l hodnot vektoru **v** (maximálně 24 hodnot pro 24-okolí) a výpočet lineárního systému například LU faktorizací. Koeficient k definuje míru poškození obrazu a platí, že $k < 1$. Koeficient h zahrnuje počet iterací výpočtu a koeficient m vyjadřuje počet metod, které jsou při rekonstrukci použité. Například při použití rekonstrukce zprava a zleva je $m = 2$.

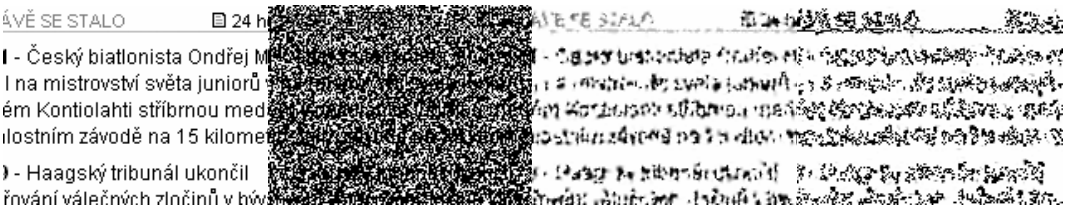
Testování metody

Obraz	S	S^2	PSNR	B	ϕ(r)	Čas [s]	Iterací
1	6,447	129,721	27,001	15360	1	2,203	34
2	8,155	173,975	25,726	4884	1	0,844	20
3	22,775	1482,671	16,420	6477	1	0,719	43
4	44,495	3585,049	12,586	3022	1	0,297	25
5	6,628	192,266	25,292	11605	1	1,641	36
6	7,418	201,957	25,078	3784	1	0,688	20
7	19,533	1408,354	16,644	5309	1	0,578	43
8	5,267	92,652	28,462	11605	1	1,625	36
9	6,482	109,619	27,732	3784	1	0,641	20
10	13,773	436,814	21,728	5309	1	0,578	43
11	47,036	7627,833	9,307	11605	1	1,625	36
12	49,610	8206,611	8,989	3784	1	0,641	20
13	68,433	13565,748	6,806	5309	1	0,594	43
14	18,138	569,579	20,575	17161	1	2,453	38
15	10,611	248,789	24,172	212880	1	30,531	125

Tabulka 9: Testování metody rekonstrukce zleva-zprava.

Obrázek 46: Ukázka postupné rekonstrukce obrazu zleva-zprava.

Pokud si všimneme nejhůře zrekonstruovaného obrazu 13 (
Obrázek 47), tak zde je problém už v samotném obrazu. Jedná se o obraz popsaný textem, kde tloušťka písmen je jeden obrazový bod. V takovém případě není možné ze znalosti okolí poškozeného bodu jednoznačně zrekonstruovat obraz. Tento obraz je zde hlavně kvůli představě co je a není možné zrekonstruovat.



Obrázek 47: Obraz s největší chybou rekonstrukce. (zleva: originální obraz, poškozený obraz, rekonstruovaný obraz, B2 obraz)

Obdobně jako metoda rekonstrukce kombinující rekonstrukci zprava a zleva lze kombinovat i rekonstrukce z jiných stran obrazu. Postupně uvedeme výsledky získané z některých kombinací stranových rekonstrukcí pro námi zvolenou skupinu testovaných obrazů a masek.

Rekonstrukce zleva-shora (LT)

Obraz	S	S^2	PSNR	B	$\phi(r)$	Čas [s]	Iterací
1	6,444	130,286	26,982	15360	1	2,172	17
2	8,172	174,328	25,717	4884	1	0,828	5
3	21,203	1222,026	17,260	6477	1	0,688	28
4	46,352	3983,480	12,128	3022	1	0,297	29
5	6,636	192,084	25,296	11605	1	1,703	14
6	7,551	210,266	24,903	3784	1	0,641	4
7	17,654	1140,293	17,561	5309	1	0,563	27
8	5,263	90,902	28,545	11605	1	1,609	14
9	6,461	107,584	27,813	3784	1	0,641	4
10	13,526	493,381	21,199	5309	1	0,578	27
11	46,886	7584,083	9,332	11605	1	1,609	14
12	49,806	8221,397	8,981	3784	1	0,641	4
13	61,017	12255,247	7,248	5309	1	0,578	27
14	18,498	644,671	20,037	17161	1	2,500	31
15	10,638	250,165	24,149	212880	1	30,250	57

Tabulka 10: Testování metody rekonstrukce zleva-shora.

Obrázek 48: Ilustrativní obrázek metody zleva-shora. (Obraz č.6)

Rekonstrukce shora-zdola (TB)

Obraz	S	S^2	PSNR	B	$\phi(r)$	Čas [s]	Iterací
1	6,413	129,092	27,022	15360	1	2,141	27
2	8,302	182,738	25,513	4884	1	0,797	19
3	21,361	1260,036	17,127	6477	1	0,688	47
4	47,960	4858,191	11,266	3022	1	0,297	41
5	6,654	196,563	25,196	11605	1	1,578	21
6	7,412	200,938	25,100	3784	1	0,625	18
7	15,104	836,075	18,908	5309	1	0,563	41
8	5,193	87,631	28,704	11605	1	1,547	21
9	6,420	105,927	27,881	3784	1	0,609	18
10	15,105	580,035	20,496	5309	1	0,563	41
11	46,880	7558,542	9,346	11605	1	1,547	21
12	48,852	8031,536	9,083	3784	1	0,609	18
13	66,822	13042,420	6,977	5309	1	0,563	41
14	18,800	705,565	19,645	17161	1	2,953	64
15	10,648	250,770	24,138	212880	1	29,547	70

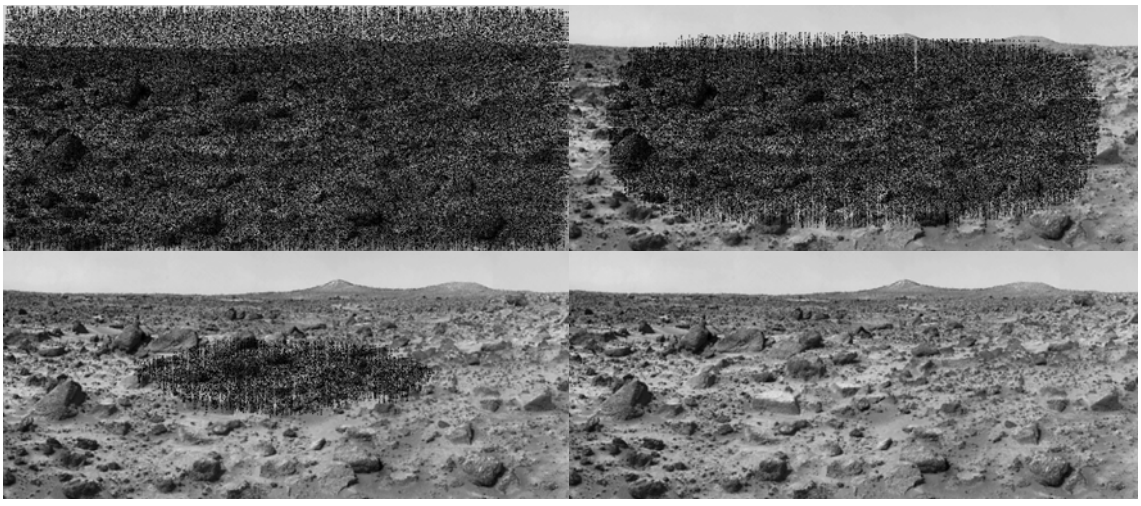
Tabulka 11: Testování metody rekonstrukce shora-zdola.

Obrázek 49: Ilustrativní obrázek průběhu metody shora-zdola. (Obraz č.5)

Rekonstrukce zleva-shora-zprava-zdola (LTRB)

Obraz	S	S^2	PSNR	B	ϕ(r)	Čas [s]	Iterací
1	6,415	129,340	27,013	15360	1	2,063	8
2	8,105	173,289	25,743	4884	1	0,797	3
3	21,749	1428,171	16,583	6477	1	0,672	14
4	39,754	3501,596	12,688	3022	1	0,297	14
5	6,677	197,080	25,184	11605	1	1,531	7
6	7,457	203,756	25,040	3784	1	0,609	2
7	16,701	1092,012	17,749	5309	1	0,547	13
8	5,221	88,586	28,657	11605	1	1,531	7
9	6,456	107,160	27,830	3784	1	0,594	2
10	12,290	381,912	22,311	5309	1	0,531	13
11	46,891	7585,733	9,331	11605	1	1,547	7
12	49,567	8188,027	8,999	3784	1	0,594	2
13	73,951	14765,988	6,438	5309	1	0,547	13
14	18,984	702,752	19,663	17161	1	2,391	16
15	10,634	249,892	24,153	212880	1	29,109	28

Tabulka 12: Testování metody rekonstrukce zleva-shora-zprava-zdola.



Obrázek 50: Ilustrativní obrázek průběhu LTRB metody. (Obraz č.15)

Rekonstrukce zleva-zprava-shora-zdola (LRTB)

Obraz	S	S^2	PSNR	B	ϕ(r)	Čas [s]	Iterací
1	6,409	129,039	27,024	15360	1	2,078	8
2	8,119	172,650	25,759	4884	1	0,781	3

3	21,635	1418,219	16,613	6477	1	0,672	14
4	39,452	3445,555	12,758	3022	1	0,281	14
5	6,670	196,487	25,197	11605	1	1,547	7
6	7,447	202,570	25,065	3784	1	0,609	3
7	16,607	1080,549	17,794	5309	1	0,547	13
8	5,232	89,131	28,631	11605	1	1,531	7
9	6,478	108,643	27,771	3784	1	0,594	3
10	12,163	370,367	22,444	5309	1	0,547	13
11	46,941	7595,547	9,325	11605	1	1,531	7
12	49,562	8183,453	9,001	3784	1	0,609	3
13	73,553	14678,047	6,464	5309	1	0,531	13
14	19,002	703,364	19,659	17161	1	2,391	16
15	10,633	249,825	24,154	212880	1	29,094	29

Tabulka 13: Testování metody rekonstrukce zleva-zprava-shora-zdola.

Průběh LRTB metody je obdobný jako u metody LTRB.

3.7.4 Scan-line metoda – oboustranná

Oboustranná scan-line metoda je modifikací jednostranné rekonstrukce z kapitoly 3.7.2. Její oboustrannost spočívá v tom, že pokud je na právě rekonstruovaném řádku nalezena díra taková, že všechny ostatní body ve směru rekonstrukce jsou díry, tak výpočet nepokračuje na další řádce, ale je nalezen konec díry a obrazový bod na konci díry je zrekonstruován.

Výhodou této modifikace je menší počet iterací pro provedení rekonstrukce celého obrazu, avšak výsledky rekonstrukce jsou obdobné jako u ostatních scan-line metod. Tato modifikace může být samozřejmě použita i pro ostatní scan-line metody.

```
Vstup: O - poškozený obraz, M - maska, k - n-okolí, ϕ - bázová funkce
Výstup: rekonstruovaný obraz

do
 for j=0:1:j<N
  for i=0:1:i<M /*po řádku zleva*/
   if(M(i,j) == díra)    /*na dané pozici je díra*/
    if(KontrolaPočtuDěrVpravo(i,j,O,M,1))
     bContinue = false;
    else
     bContinue = true;

    /* rekonstrukce je zastavena na všechny obrazové body mezi prvním a
       posledním bodem díry
    */
    if(bContinue)
     v = OkolíBodu(i,j,O,k);
     B = VytvořMatici(v,ϕ);
     b = VektorPravychStran(v);
     lambda = RešSystém(B,b);
     O(i,j) = HodnotaBodu(i,j,lambda,v,ϕ);
     ZrušDíru(M,i,j);

  end;
 end;
end;
while(!JsouJestěNějakéDíry(M))
```

Algoritmus 7: Scan-line metoda rekonstrukce koncového a počátečního obrazového bodu delší díry.

Paměťová náročnost algoritmu je shodná s předchozími metodami. Výraz algoritmické složitosti je tedy

$$O\left(k\left(hnl^4\right)\right), \tag{3}$$

kde n je hlavní výpočet přes celý obraz a l^4 v sobě zahrnuje vytvoření lineárního systému z l hodnot vektoru **v** (maximálně 24 hodnot pro 24-okolí) a výpočet lineárního systému například LU faktorizací. Koeficient k definuje míru poškození obrazu a platí, že $k < 1$. Koeficient h zahrnuje počet iterací výpočtu.

Testování metody

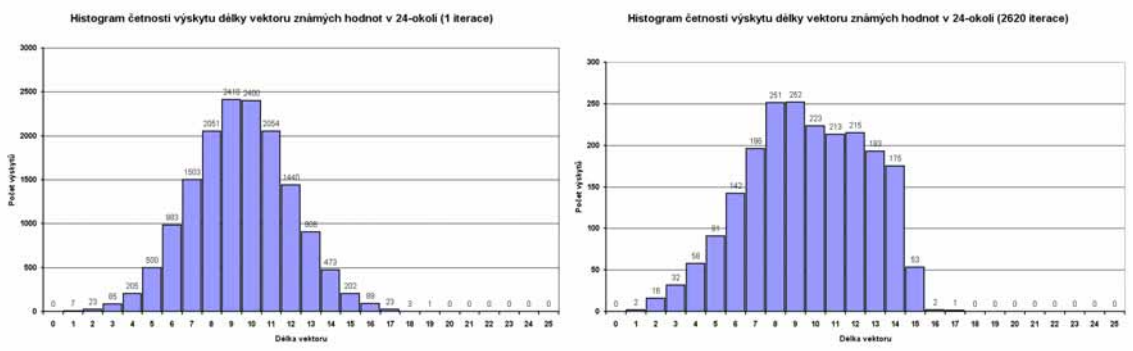
Obraz	S	S^2	PSNR	B	$\phi(r)$	Čas [s]	Iterací
1	6,491	132,394	26,912	15360	1	2,188	9
2	8,161	171,437	25,790	4884	1	0,906	8
3	18,944	1007,854	18,097	6477	1	1,016	34
4	41,484	3703,784	12,444	3022	1	0,656	35
5	6,628	191,321	25,313	11605	1	1,703	10
6	7,515	207,503	24,961	3784	1	0,719	8
7	15,092	873,088	18,720	5309	1	0,906	34
8	5,242	89,660	28,605	11605	1	1,703	10
9	6,468	108,550	27,775	3784	1	0,719	8
10	12,472	396,909	22,144	5309	1	0,875	34
11	46,924	7601,281	9,322	11605	1	1,703	10
12	49,479	8192,463	8,997	3784	1	0,703	8
13	59,371	11524,122	7,515	5309	1	0,891	34
14	17,179	509,869	21,056	17161	1	2,578	9
15	10,561	245,902	24,223	212880	1	30,297	18

Tabulka 14: Testování metody oboustranné rekonstrukce.

Obrázek 51: Ilustrativní obrázek průběhu oboustranné metody. (Obraz č.1)

3.7.5 Maximální informace v k-okolí

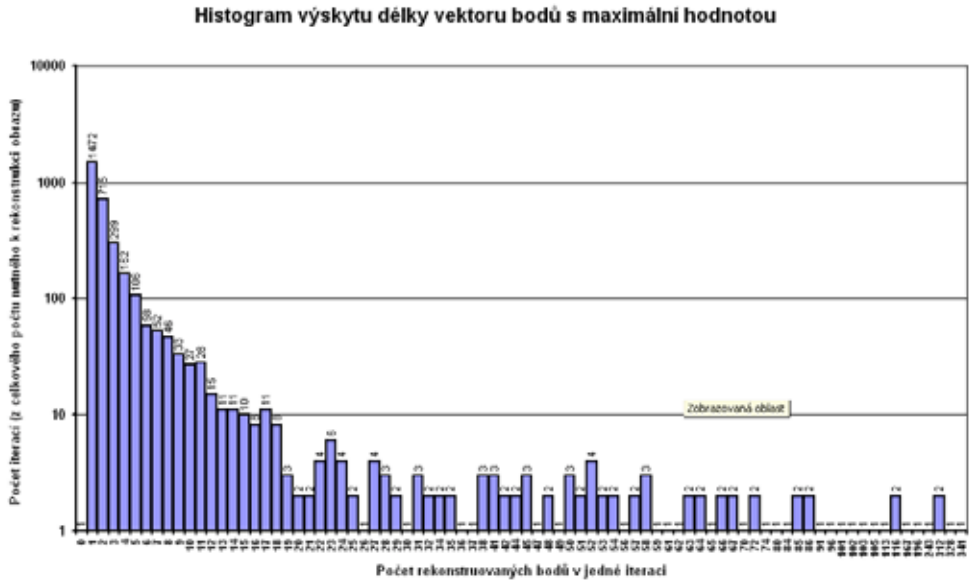
Tato metoda je založena na výběru oblastí v obrazu, které mají největší počet známých hodnot v n-okolí. To znamená, že jsme schopni dosáhnout maximální možné informace, abychom co nejlépe dopočetli chybějící hodnotu poškozeného obrazového bodu. Tato metoda by měla být z ohledu teorie informace nejlepší v rekonstrukci, avšak v některých případech tomu tak není.



Obrázek 52: Histogramy počtu výskytů délky vektoru známých hodnot v 24-okolí (vlevo: vstupní obraz, vpravo: 2620 iterace rekonstrukce)

Metoda pracuje na postupné rekonstrukci, kdy jsou v každé iteraci zrekonstruovány pouze body u nichž známe největší počet hodnot v okolí. Na Obrázek 52 je histogram udávající kolik známe hodnot v okolí u kolika bodů. Z levého obrázku je vidět, že existuje jeden bod, u něhož známe 19 hodnot v okolí. V první iteraci bude tedy zrekonstruován tento obrazový bod. Na pravém obrázku je vidět vývoj histogram po několika iteracích. V této iteraci bude provedena rekonstrukce jediného bodu u něhož známe 17 hodnot v okolí.

Obrázek 53: Rekonstrukce obrazu k němuž je uveden histogram. (vlevo: vstupní obraz, uprostřed: 1310 iterace rekonstrukce, vpravo: 2620 iterace rekonstrukce, Obraz č.1)



Obrázek 54: Histogram počtu výskytu shluků určitých délek při rekonstrukci. (Obraz č.1)

Z uvedených histogramů je vidět, že počet iterací bude vysoký, ale neznamená to, že bude v každé iteraci zrekonstruovaný pouze jeden bod. Jak je možné vidět na histogramu výše (Obrázek 54), tak v některých případech bude během jedné iterace metody nalezeno několik bodů s maximálním počtem známých bodů v okolí. Všimněme si například, že byla během rekonstrukce jedna iterace, kdy bylo nalezeno 341 bodů se stejným počtem známých hodnot obrazových bodů v n-okolí poškozeného bodu.

Algoritmus metody má několik částí. První z nich je funkce `MaxVectorSize()`, která projde obraz masky a z okolí všech bodů masky zjistí jaké je maximum známých hodnot v jejich okolí. Algoritmická složitost této části je O(n), neboť hledání vhodného bodu k rekonstrukci je provedeno průchodem přes všechny obrazové body ($n = M \times N$). Pokud je obrazový bod díra, tak jsou spočteny všechny známé hodnoty v jeho okolí. Dolní hranice algoritmické složitosti je O(k), kde k je počet děr v masce.

```
Vstup: O - poškozený obraz, M - maska, k - n-okolí, ϕ - bázová funkce
Výstup: rekonstruovaný obraz

do
 iMax = MaxVectorSize(M);
 pix = VratSeznamBoduSMaxDelkouVektoru(O,iMax);
 for i=0:1:i<pix.počet
  if(M(pix[i].i,pix[i].j) == díra)    /*na dané pozici je díra*/
   v = OkolíBodu(pix[i].i,pix[i].j,O,k);
   B = VytvořMatici(v,ϕ);
   b = VektorPravychStran(v);
   lambda = RešSystém(B,b);
   O(pix[i].i,pix[i].j) = HodnotaBodu(pix[i].i,pix[i].j,lambda,v,ϕ);
   ZrušDíru(M,i,j);
  end;
 end;
while(!JsouJestěNějakéDíry(M))
```

Algoritmus 8: Maximální informace z k-okolí.

Další částí je funkce `VratSeznamBoduSMaxDelkouVektoru(O)`, která vrací seznam bodů, jež mají v okolí `iMax` známých hodnot. Funkce má také algoritmickou složitost O(n). Algoritmická složitost celého algoritmu je tedy

$$O\left(k\left(n^3 l^4 h\right)\right), \tag{4}$$

kde n^3 je hlavní výpočet přes celý obraz, zjištění maximální délky vektoru a nalezení všech bodů s touto maximální délkou. l^4 v sobě zahrnuje vytvoření lineárního systému z l hodnot vektoru **v** (maximálně 24 hodnot pro 24-okolí) a výpočet lineárního systému například LU faktorizací. Výpočet lineárního systému se provede h-krát, kde h je počet nalezených bodů s maximálním okolím. Koeficient k definuje míru poškození obrazu a platí, že $k < 1$.

Algoritmická složitost celé metody je poměrně vysoká. Zcela určitě je zde prostor pro urychlení algoritmu například použitím metody řazení nebo stromu v případě zjišťování maximální délky vektoru a následné nalezení všech hodnot s touto délkou. Tyto dva kroky by se daly sloučit. Nám zde šlo o ověření domněnky, že tato metoda bude nejlépe provádět rekonstrukci.

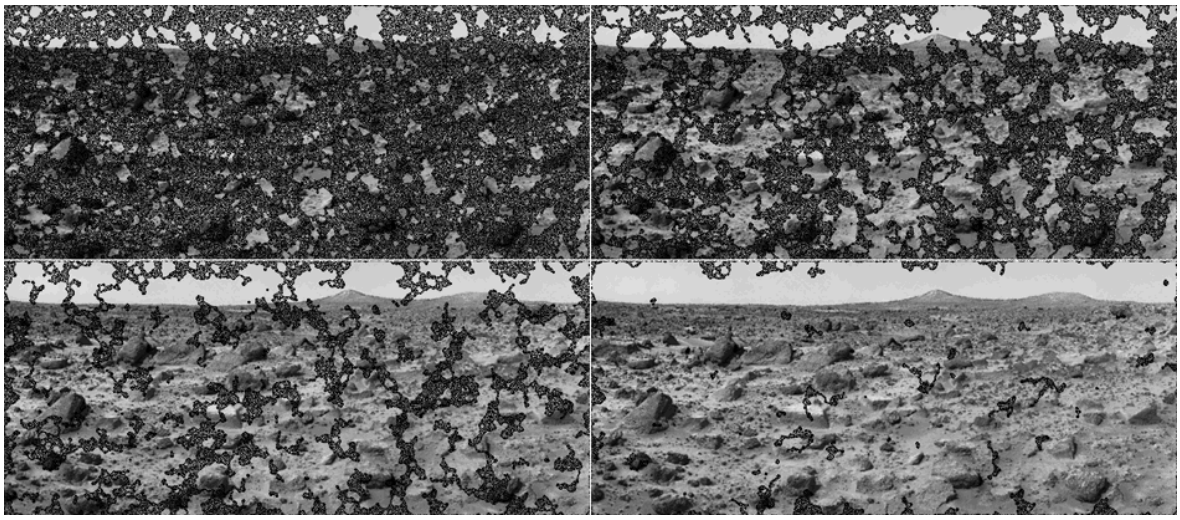
Paměťová náročnost celého algoritmu je minimální stejně jako v předchozích metodách rekonstrukce. Je nutné ukládat pouze obraz originální, obraz masky a výsledný obraz. Ostatní paměťové požadavky jsou zanedbatelné.

Testování metody

Obraz	S	S^2	PSNR	B	$\phi(r)$	Čas [s]	Iterací
1	6,528	133,066	26,890	15360	1	58,078	3192
2	8,165	170,572	25,812	4884	1	1,297	82
3	18,359	922,155	18,483	6477	1	11,625	2093
4	42,267	3777,467	12,359	3022	1	0,938	403
5	6,633	192,944	25,276	11605	1	32,453	2359
6	7,479	202,636	25,064	3784	1	1,016	86
7	14,601	754,531	19,354	5309	1	4,891	1163
8	5,197	86,634	28,754	11605	1	32,688	2359
9	6,454	108,357	27,782	3784	1	1,016	86
10	12,098	380,270	22,330	5309	1	4,875	1163
11	47,043	7585,566	9,331	11605	1	32,422	2359
12	49,199	8115,553	9,038	3784	1	1,000	86
13	61,433	11503,294	7,523	5309	1	4,875	1163
14	16,562	465,584	21,451	17161	1	12,109	467
15	10,626	249,212	24,165	212880	1	4532,250	18953

Tabulka 15: Testování metody maximální délky vektoru.

Z výše uvedené tabulky si můžete všimnout, že pro většinu poškozených obrazů je doba výpočtu v akceptovatelných mezích. Pouze v případě velkého poškození roztroušenými body je doba výpočtu jedna hodina už nepřijatelná (obraz č.15). Je nutné taktéž poznamenat, že v případě rekonstrukce barevného obrazu by se hodnoty ztrojnásobili neboť je nutné provést rekonstrukci pro RGB složky obrazu. Zde provádíme rekonstrukce pouze jasové složky obrazu.



Obrázek 55: Rekonstrukce obrazu č.15 metodou maximální informace v k-okolí. Rekonstruovaný obraz po dvou, šesti, deseti a šestnácti tisících iteracích.

3.8 Detekce poškozených obrazových bodů

Jak již bylo poznamenáno dříve, tak se tato práce nezabývá metodami detekce poškozené části obrazu. Přesto bychom se měli zmínit alespoň o základních metodách.

Nejjednodušší, ale zároveň nejpracnější metoda je ruční označení poškozených oblastí. Tento způsob je pracný, ale zřejmě nenahraditelný. Neexistuje dokonalá automatická metoda, která by byla schopna detekovat všechna poškození v obraze.

Nejčastěji jsou tedy používány poloautomatické metody, kdy uživatel klikne do oblasti, která je poškozením (např. dírou) a pomocí různých algoritmů plnění je oblast s konstantní barvou nebo jasem označena. Lze samozřejmě definovat meze intenzity barvy značící poškození. Tímto už se dostáváme k metodě prahování, kdy je definovaná horní a dolní mez intenzity barvy a jsou automaticky vybrány obrazové body s intenzitou v tomto intervalu.

Obrázek 56: Označení poškozené části obrazu a) automaticky například vyplněním oblasti se stejnou intenzitou obrazových bodů b) ručně vytvořená maska.

Obrázek 56 ukazuje použití poloautomatické metody, která vyplňuje oblast konstantní barvy a metodu ruční definice masky. V případě definice masky pro Obrázek 56b je velmi obtížně určit prahovou hodnotu pro poškození, aby bylo možné využití poloautomatické metody. V našem případě máme masky definované konstantní jasovou hodnotou.

3.9 Porovnání metod

V kapitole 3.7 jsme si uvedli několik námi navržených metod rekonstrukce poškozených obrazů RBF metodou. Dále jsme uvedli výsledky rekonstrukce různě poškozených obrazů, jejichž seznam naleznete v příloze E. V této kapitole si provedeme porovnání všech uvedených metod na kvalitu rekonstrukce poškozených obrazů. Postupně projdeme jednotlivé obrazy a zhodnotíme kvalitu rekonstrukce jak vizuálně, tak i MSE (*mean square error*) chybu a PSNR (*peak signal-to-noise ratio*) chybu. Ještě poznamenejme, že MSE chyba by měla být co nejmenší a PSNR co největší.

Uvedeme si ještě označení jednotlivých metod, které budeme používat v následujících tabulkách. Označení chybových obrazů jsme si již uvedli v Tabulka 3.

Označení metody	Metoda rekonstrukce	Kapitola
OI	přímá (jedna iterace)	3.7.1
L	zleva	3.7.2
LR	zleva-zprava	3.7.3
LT	zleva-shora	3.7.3
TB	shora-zdola	3.7.3
LTRB	zleva-shora-zprava-zdola	3.7.3
LRTB	zleva-zprava-shora-zdola	3.7.3
OH	oboustranná	3.7.4
MAX	max. informace v k-okolí	3.7.5

Tabulka 16: Označení metod rekonstrukce.

Nejlepší výsledky rekonstrukce obrazu jsou v každé tabulce zvýrazněny. Podstatné jsou hodnoty S a S^2, kde čím menší je hodnota, tím je rekonstrukce lepší, a pak hodnota PSNR, kde je to naopak.

3.9.1 Výsledky rekonstrukce jednotlivých metod

Obraz č.1

a) b) c)

Obrázek 57: Rekonstrukce obrazu č.1 s 60% poškozením TPS bázovou funkcí a TB metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	6,455	130,766	26,966	15360	1	2,063	1
L	6,437	129,948	26,993	15360	1	2,172	67
LR	6,447	129,721	27,001	15360	1	2,203	34
LT	6,444	130,286	26,982	15360	1	2,172	17
MAX	6,528	133,066	26,890	15360	1	58,078	3192
TB	6,413	129,092	27,022	15360	1	2,141	27
LTRB	6,415	129,340	27,013	15360	1	2,063	8
LRTB	6,409	129,039	27,024	15360	1	2,078	8
OH	6,491	132,394	26,912	15360	1	2,188	9

Tabulka 17: Porovnání metod rekonstrukce na obrazu č.1.

Nejlepší rekonstrukce byla provedena metodou LTRB a hned potom TB (nejlepší hodnoty jsou v tabulce zvýrazněny). Pokud si však všimneme výsledků i u jiných metod tak jsou všechny velmi podobné. Pouze časové výsledky pro metodu MAX vektoru jsou větší, což je způsobeno velkým počtem iterací a opakovaným vytvářením vektoru bodů, které mají největší počet známých hodnot v okolí.

Obraz č.2

a) b) c)

Obrázek 58: Rekonstrukce obrazu č.2 s 19% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	8,173	173,050	25,749	4884	1	0,781	1
L	8,148	172,640	25,759	4884	1	0,844	41
LR	8,155	173,975	25,726	4884	1	0,844	20
LT	8,172	174,328	25,717	4884	1	0,828	5
MAX	8,165	170,572	25,812	4884	1	1,297	82
TB	8,302	182,738	25,513	4884	1	0,797	19
LTRB	8,105	173,289	25,743	4884	1	0,797	3
LRTB	8,119	172,650	25,759	4884	1	0,781	3
OH	8,161	171,437	25,790	4884	1	0,906	8

Tabulka 18: Porovnání metod rekonstrukce na obrazu č.2.

Výsledky rekonstrukce obrazu poškozeného inpaintingem vycházejí nejlépe pro metodu MAX. Metoda MAX je sice také nejpomalejší, ale není to nijak významné zpomalení.

Obraz č.3

a) b) c)

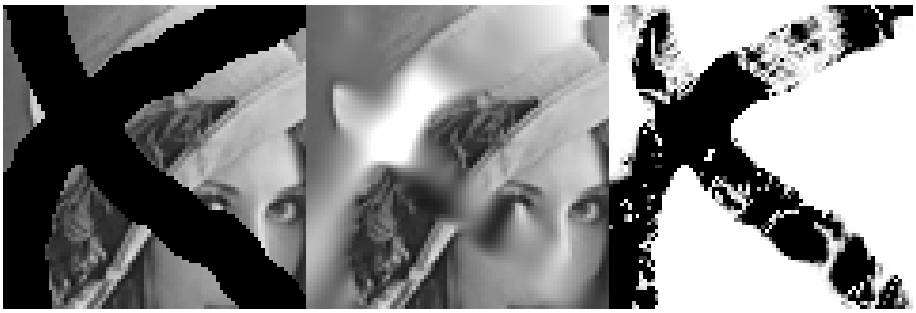
Obrázek 59: Rekonstrukce obrazu č.3 s 25% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	19,036	929,714	18,447	6477	1	0,625	1
L	21,100	1161,591	17,480	6477	1	0,734	85
LR	22,775	1482,671	16,420	6477	1	0,719	43
LT	21,203	1222,026	17,260	6477	1	0,688	28
MAX	18,359	922,155	18,483	6477	1	11,625	2093
TB	21,361	1260,036	17,127	6477	1	0,688	47
LTRB	21,749	1428,171	16,583	6477	1	0,672	14
LRTB	21,635	1418,219	16,613	6477	1	0,672	14
OH	18,944	1007,854	18,097	6477	1	1,016	34

Tabulka 19: Porovnání metod rekonstrukce na obrazu č.3.

V rekonstrukci obrazu poškozeného náhodnými čárami byla nejúspěšnější metoda MAX. Za zmínku stojí určitě PSNR chyba u OI rekonstrukce, která je velmi dobrá v porovnání s ostatními metodami. Metoda MAX je opět nejpomalejší.

Obraz č.4



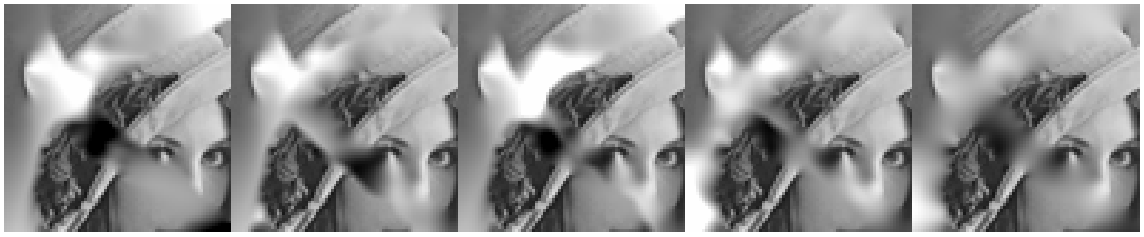
a) b) c)

Obrázek 60: Rekonstrukce obrazu č.4 s 47% poškozením TPS bázovou funkcí a OI metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	40,541	3325,686	12,912	3022	1	0,266	1
L	45,173	3835,641	12,292	3022	1	0,328	49
LR	44,495	3585,049	12,586	3022	1	0,297	25
LT	46,352	3983,480	12,128	3022	1	0,297	29
MAX	42,267	3777,467	12,359	3022	1	0,938	403
TB	47,960	4858,191	11,266	3022	1	0,297	41
LTRB	39,754	3501,596	12,688	3022	1	0,297	14
LRTB	39,452	3445,555	12,758	3022	1	0,281	14
OH	41,484	3703,784	12,444	3022	1	0,656	35

Tabulka 20: Porovnání metod rekonstrukce na obrazu č.4.

V tom velmi značném poškození si povšimněte chyby rekonstrukce ve středu poškození. Tato chyba rekonstrukce se vykytuje u většiny z uvedených metod. Jde o to, že se z okolí poškození šíří světlé pixely do poškozené části. Rekonstrukce z ostatních metod jsou uvedeny níže (Obrázek 61).



Obrázek 61: Rekonstrukce L, LR, LT, MAX a globální metodou.

Rekonstrukce globální metodou je uvedena v Tabulka 4 a její výsledky jsou lepší než jakákoliv lokální rekonstrukce uvedená v Tabulka 20.

Obraz č.5

a) b) c)

Obrázek 62: Rekonstrukce obrazu č.5 s 60% poškozením TPS bázovou funkcí a LT metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	6,690	195,388	25,222	11605	1	1,500	1
L	6,616	191,083	25,319	11605	1	1,656	71
LR	6,628	192,266	25,292	11605	1	1,641	36
LT	6,636	192,084	25,296	11605	1	1,703	14
MAX	6,633	192,944	25,276	11605	1	32,453	2359
TB	6,654	196,563	25,196	11605	1	1,578	21
LTRB	6,677	197,080	25,184	11605	1	1,531	7
LRTB	6,670	196,487	25,197	11605	1	1,547	7
OH	6,628	191,321	25,313	11605	1	1,703	10

Tabulka 21: Porovnání metod rekonstrukce na obrazu č.5.

Obraz obsahuje velké množství hran což je vidět výše na rozdílovém obrazeW2 (Obrázek 62c).

Obraz č.6

a) b) c)

Obrázek 63: Rekonstrukce obrazu č.6 s 20% poškozením TPS bázovou funkcí a TB metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	7,554	210,206	24,904	3784	1	0,594	1
L	7,506	209,647	24,916	3784	1	0,750	41
LR	7,418	201,957	25,078	3784	1	0,688	20
LT	7,551	210,266	24,903	3784	1	0,641	4
MAX	7,479	202,636	25,064	3784	1	1,016	86
TB	7,412	200,938	25,100	3784	1	0,625	18
LTRB	7,457	203,756	25,040	3784	1	0,609	2
LRTB	7,447	202,570	25,065	3784	1	0,609	3
OH	7,515	207,503	24,961	3784	1	0,719	8

Tabulka 22: Porovnání metod rekonstrukce na obrazu č.6.

Obraz č.7

a) b) c)

Obrázek 64: Rekonstrukce obrazu č.7 s 27% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	16,519	952,787	18,341	5309	1	0,516	1
L	20,752	1568,646	16,176	5309	1	0,594	85
LR	19,533	1408,354	16,644	5309	1	0,578	43
LT	17,654	1140,293	17,561	5309	1	0,563	27
MAX	14,601	754,531	19,354	5309	1	4,891	1163
TB	15,104	836,075	18,908	5309	1	0,563	41
LTRB	16,701	1092,012	17,749	5309	1	0,547	13
LRTB	16,607	1080,549	17,794	5309	1	0,547	13
OH	15,092	873,088	18,720	5309	1	0,906	34

Tabulka 23: Porovnání metod rekonstrukce na obrazu č.7.

Obraz č.8

a) b) c)

Obrázek 65: Rekonstrukce obrazu č.8 s 60% poškozením TPS bázovou funkcí a LR metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	5,281	90,595	28,560	11605	1	1,469	1
L	5,296	91,691	28,508	11605	1	1,641	71
LR	5,267	92,652	28,462	11605	1	1,625	36
LT	5,263	90,902	28,545	11605	1	1,609	14
MAX	5,197	86,634	28,754	11605	1	32,688	2359
TB	5,193	87,631	28,704	11605	1	1,547	21
LTRB	5,221	88,586	28,657	11605	1	1,531	7
LRTB	5,232	89,131	28,631	11605	1	1,531	7
OH	5,242	89,660	28,605	11605	1	1,703	10

Tabulka 24: Porovnání metod rekonstrukce na obrazu č.8.

Obraz č.9

a) b) c)

Obrázek 66: Rekonstrukce obrazu č.9 s 20% poškozením TPS bázovou funkcí a TB metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	6,465	107,881	27,801	3784	1	0,594	1
L	6,489	108,811	27,764	3784	1	0,656	41
LR	6,482	109,619	27,732	3784	1	0,641	20
LT	6,461	107,584	27,813	3784	1	0,641	4
MAX	6,454	108,357	27,782	3784	1	1,016	86
TB	6,420	105,927	27,881	3784	1	0,609	18
LTRB	6,456	107,160	27,830	3784	1	0,594	2
LRTB	6,478	108,643	27,771	3784	1	0,594	3
OH	6,468	108,550	27,775	3784	1	0,719	8

Tabulka 25: Porovnání metod rekonstrukce na obrazu č.9.

Obraz č.10

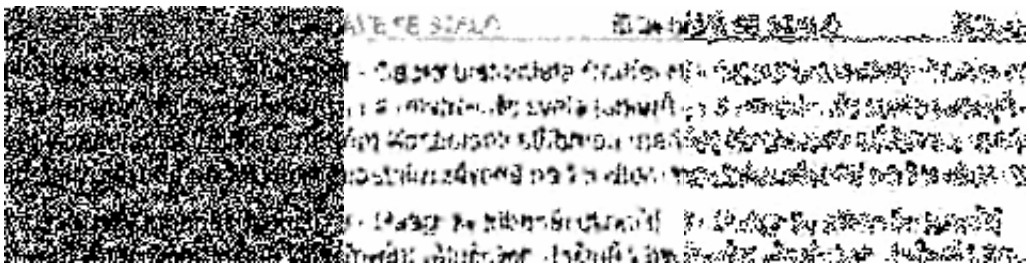
a) b) c)

Obrázek 67: Rekonstrukce obrazu č.10 s 28% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	14,672	576,893	20,520	5309	1	0,516	1
L	15,450	624,572	20,175	5309	1	0,594	85
LR	13,773	436,814	21,728	5309	1	0,578	43
LT	13,526	493,381	21,199	5309	1	0,578	27
MAX	12,098	380,270	22,330	5309	1	4,875	1163
TB	15,105	580,035	20,496	5309	1	0,563	41
LTRB	12,290	381,912	22,311	5309	1	0,531	13
LRTB	12,163	370,367	22,444	5309	1	0,547	13
OH	12,472	396,909	22,144	5309	1	0,875	34

Tabulka 26: Porovnání metod rekonstrukce na obrazu č.10.

Obraz č.11



a) b) c)

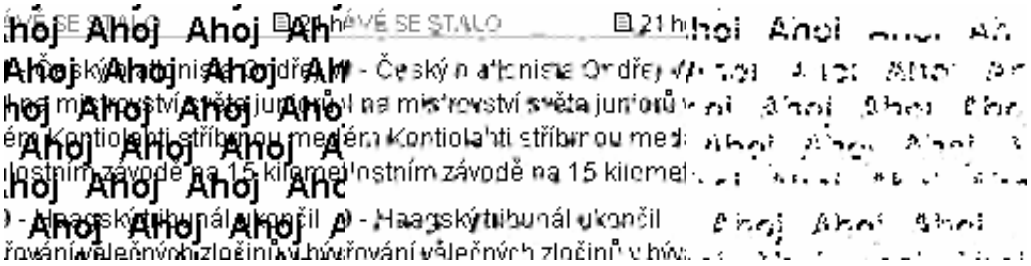
Obrázek 68: Rekonstrukce obrazu č.11 s 60% poškozením TPS bázovou funkcí a LR metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	47,011	7596,265	9,325	11605	1	1,469	1
L	47,003	7599,810	9,323	11605	1	1,672	71
LR	47,036	7627,833	9,307	11605	1	1,625	36
LT	46,886	7584,083	9,332	11605	1	1,609	14
MAX	47,043	7585,566	9,331	11605	1	32,422	2359
TB	46,880	7558,542	9,346	11605	1	1,547	21
LTRB	46,891	7585,733	9,331	11605	1	1,547	7
LRTB	46,941	7595,547	9,325	11605	1	1,531	7
OH	46,924	7601,281	9,322	11605	1	1,703	10

Tabulka 27: Porovnání metod rekonstrukce na obrazu č.11.

Rekonstrukce textu poškozeného šumem je i při znalosti poškozených obrazových bodů velmi obtížné. Hlavně v uvedeném případě, kdy se tloušťka písma blíží k velikosti jednoho obrazového bodu. Při rekonstrukci mají pak na výslednou hodnotu obrovský vliv známé obrazové body s bílou barvou (platí pro náš případ) a výsledná hodnota rekonstruovaného bodu bude s největší pravděpodobností doplněna špatně. Výsledkem je pak nečitelný rekonstruovaný text. Korektní rekonstrukce takto poškozeného obrazu je prakticky nemožná.

Obraz č.12



a) b) c)

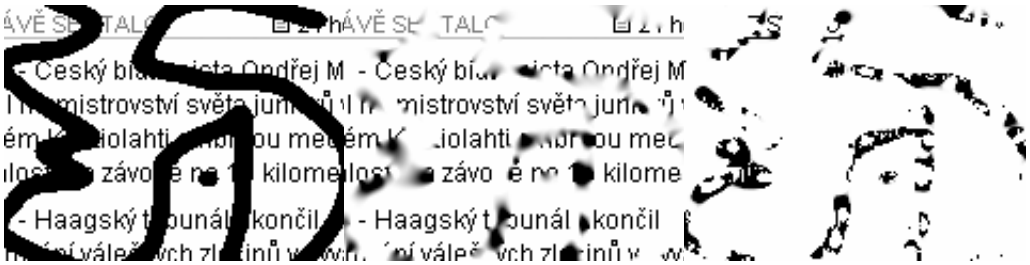
Obrázek 69: Rekonstrukce obrazu č.12 s 20% poškozením TPS bázovou funkcí a TB metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	49,540	8201,704	8,992	3784	1	0,594	1
L	49,508	8216,054	8,984	3784	1	0,641	41
LR	49,610	8206,611	8,989	3784	1	0,641	20
LT	49,806	8221,397	8,981	3784	1	0,641	4
MAX	49,199	8115,553	9,038	3784	1	1,000	86
TB	48,852	8031,536	9,083	3784	1	0,609	18
LTRB	49,567	8188,027	8,999	3784	1	0,594	2
LRTB	49,562	8183,453	9,001	3784	1	0,609	3
OH	49,479	8192,463	8,997	3784	1	0,703	8

Tabulka 28: Porovnání metod rekonstrukce na obrazu č.12.

V tomto případě, kdy poškození textu není příliš veliké by se spíše hodila jiná metoda rekonstrukce založená například na vyhledávání vzorů (jednotlivých písmen) v obraze.

Obraz č.13



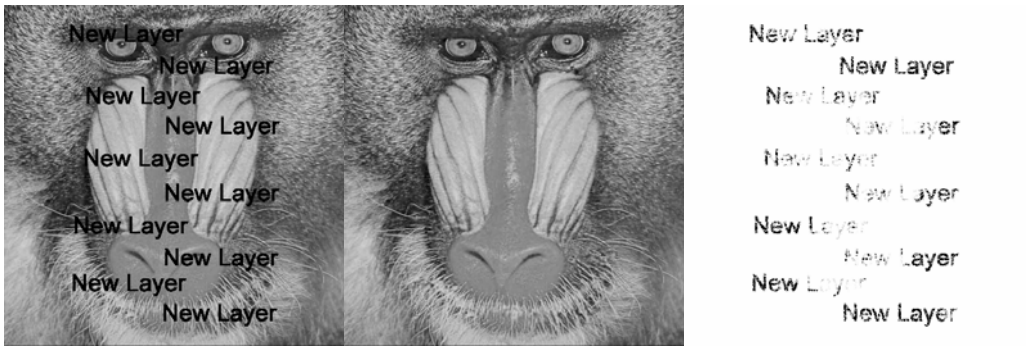
a) b) c)

Obrázek 70: Rekonstrukce obrazu č.13 s 28% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	65,684	13042,939	6,977	5309	1	0,516	1
L	57,965	11848,340	7,394	5309	1	0,609	85
LR	68,433	13565,748	6,806	5309	1	0,594	43
LT	61,017	12255,247	7,248	5309	1	0,578	27
MAX	61,433	11503,294	7,523	5309	1	4,875	1163
TB	66,822	13042,420	6,977	5309	1	0,563	41
LTRB	73,951	14765,988	6,438	5309	1	0,547	13
LRTB	73,553	14678,047	6,464	5309	1	0,531	13
OH	59,371	11524,122	7,515	5309	1	0,891	34

Tabulka 29: Porovnání metod rekonstrukce na obrazu č.13.

Obraz č.14



a) b) c)

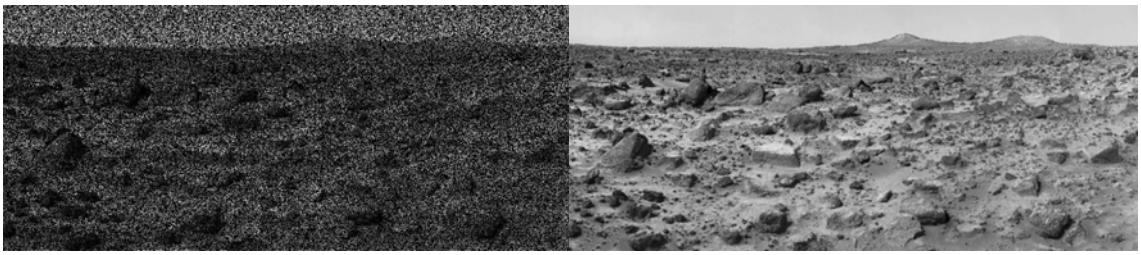
Obrázek 71: Rekonstrukce obrazu č.14 s 6.5% poškozením TPS bázovou funkcí a MAX metodou. (a) poškozený obraz, b) rekonstruovaný obraz, c) rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	17,467	540,600	20,802	17161	1	2,047	1
L	18,153	586,088	20,451	17161	1	2,484	75
LR	18,138	569,579	20,575	17161	1	2,453	38
LT	18,498	644,671	20,037	17161	1	2,500	31
MAX	16,562	465,584	21,451	17161	1	12,109	467
TB	18,800	705,565	19,645	17161	1	2,953	64
LTRB	18,984	702,752	19,663	17161	1	2,391	16
LRTB	19,002	703,364	19,659	17161	1	2,391	16
OH	17,179	509,869	21,056	17161	1	2,578	9

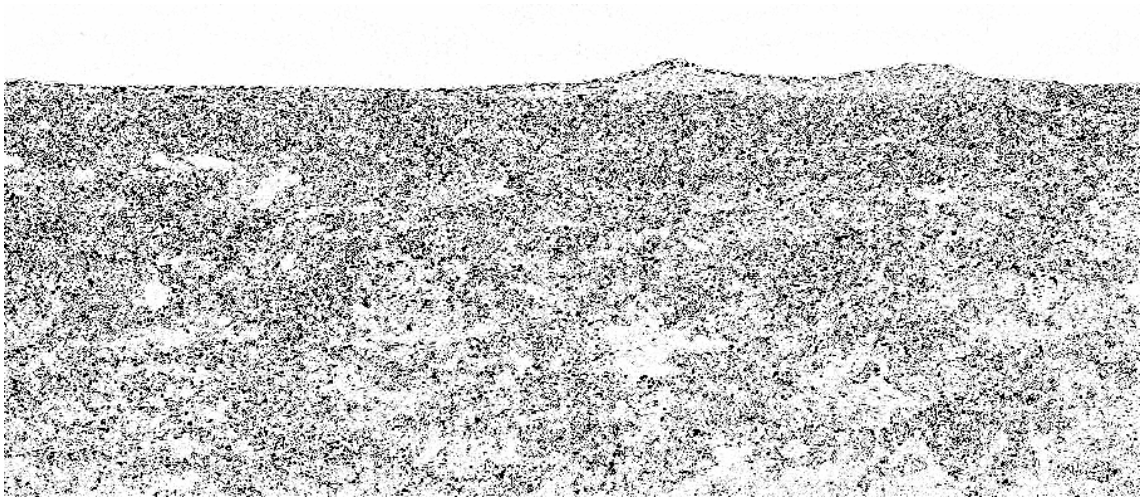
Tabulka 30: Porovnání metod rekonstrukce na obrazu č.14.

Poškozený obraz, který má větší rozlišení a to 512 x 512 obrazových bodů je velmi kvalitně rekonstruován. Původní poškození není na výsledném obrazu vůbec patrné.

Obraz č.15



Obrázek 72: Rekonstrukce obrazu č.15 s 60% poškozením TPS bázovou funkcí a LR metodou. (vlevo - poškozený obraz, vpravo - rekonstruovaný obraz)



Obrázek 73: Rekonstrukce obrazu č.15 s 60% poškozením TPS bázovou funkcí a LR metodou. (rozdílový obraz W2)

Metoda	S	S^2	PSNR [dB]	B	$\phi(r)$	Čas [s]	Iterací
OI	10,646	250,660	24,140	212880	1	27,750	1
L	10,622	249,220	24,165	212880	1	30,813	249
LR	10,611	248,789	24,172	212880	1	30,531	125
LT	10,638	250,165	24,149	212880	1	30,250	57
MAX	10,626	249,212	24,165	212880	1	4532,250	18953
TB	10,648	250,770	24,138	212880	1	29,547	70
LTRB	10,634	249,892	24,153	212880	1	29,109	28
LRTB	10,633	249,825	24,154	212880	1	29,094	29
OH	10,561	245,902	24,223	212880	1	30,297	18

Tabulka 31: Porovnání metod rekonstrukce na obrazu č.15.

Přesto, že obraz č.15 je poměrně značně poškozen, tak jsme dostali dosti kvalitní rekonstruovaný obraz. V podstatě nám stačilo znát 40% obrazových bodů, abychom dokázali obraz opravit. Z výsledného obrazu je možné bez problému rozpoznat jednotlivé objekty.

3.9.2 Porovnání metod rekonstrukce

Pro lepší porovnání metod je níže uvedena přehledová tabulka metod s nejlepšími výsledky pro daný obraz a daný způsob porovnání chyby rekonstrukce (Tabulka 32).

Metody rekonstrukce je nutné porovnávat z několika kritérii. Prvním z nich by mohla být kvalita rekonstrukce. Nejlepší metodou z hlediska kvality rekonstrukce je MAX metoda. Tato metoda poskytuje velmi kvalitní výsledek pro různě poškozené obrazy a přes to, že není nejlepší při některých rekonstrukcích, tak její výsledky nejsou příliš rozdílné od metody nejlepší. Nejlépe dokáže metoda opravit jednoduché škrábance nebo poškození textem. To je hlavně kvůli dostatečnému počtu známých hodnot v okolí rekonstruovaného obrazového bodu. Nevýhodou této metody je její rychlost. Metoda je pomalejší hlavně v případech velkého a rozsáhlého poškození přes celý obraz. Pro takto poškozené obrazy je vhodnější použít metodu rekonstrukce z více stran. Konkrétně metody rekonstrukce ze dvou stran prováděli nejlepší rekonstrukci obrazů poškozených šumem popřípadě textem. Nejlepší byla metoda TB při použití na text nebo šum. V případě poškození textem je to spíš způsobeno charakterem textu, který je psán horizontálně. V případě jiné orientace textu se dostávají ke slovu i jiné stranové metody. Všechny stranové metody jsou dostatečně rychlé pro všechny druhy poškození. Pokud bychom měli hledět na rychlost rekonstrukce, tak nejrychlejší byla jedno průchodová metoda OI, neboť se při více průchodech zvětší režie rekonstrukce. Ostatní metody se liší v časech jen minimálně a občas mají i shodnou dobu rekonstrukce jako OI metoda. Nejhorší je již zmíněná MAX metoda, která se občas liší i v řádech desítek sekund.

Obraz	S	S^2	PSNR	$\phi(r)$	Poškození
1	LRTB	LRTB	LRTB	1	šum
2	LTRB	MAX	MAX	1	text
3	MAX	MAX	MAX	1	škrábance
4	LRTB	OI	OI	1	škrábance
5	L	L	L	1	šum
6	TB	TB	TB	1	text
7	MAX	MAX	MAX	1	škrábance
8	TB	MAX	MAX	1	šum
9	TB	TB	TB	1	text
10	MAX	LRTB	LRTB	1	škrábance
11	TB	TB	TB	1	šum
12	TB	TB	TB	1	text
13	L	MAX	MAX	1	škrábance
14	MAX	MAX	MAX	1	text
15	OH	OH	OH	1	šum

Tabulka 32: Vyhodnocení nejlepší metody rekonstrukce pro daný obraz.

Z výsledků rekonstrukce vyplývají i oblasti použití jednotlivých metod. Pokud by se jednalo o rekonstrukci obrazu poškozeného textem, tak by bylo nejlepší použít jednu ze stranových metod se směrem rekonstrukce ve směru textu nebo MAX metodu. Pro rekonstrukci obrazu poškozeného rovnoměrně rozloženým šumem je vhodné použít stranovou metodu. Metoda rekonstrukce by měla pracovat ve směru širší strany obrazu. Metody jsou časově nenáročné a většinou poskytují lepší rekonstrukci než jedno průchodová OI metoda za téměř shodný čas. Pokud bychom požadovali nejlepší výsledek rekonstrukce v případě poškození škrábanci, tak by bylo nejlepší použít časově náročnější MAX metodu. Jinak můžeme použít libovolnou stranovou metodu. MAX metoda je sázka na jistotu za cenu delší doby rekonstrukce.

3.10 Porovnání s existujícími metodami rekonstrukce poškozených obrazů

Jak jsme se zmínili v úvodu kapitoly 3, tak jsou metody většinou rozděleny na metody opravující poškození *inpaintingem* (textem) a metody opravující poškození šumem nebo škrábanci. Metoda založená na RBF se dokáže vypořádat se všemi uvedenými typy poškození. V následujících kapitolách provedeme porovnání s Bertalmiho metodou zvládající inpainting a jednoduché škrábance. Potom provedeme porovnání s metodou založenou na CSRBF, která zvládá taktéž opravit všechny druhy poškození.

3.10.1 Inpainting, škrábance, šum

Významnou prací v rekonstrukci obrazu poškozeného textem je Bertalmiho metoda založená na parciálních diferenciálních rovnicích. Vzhledem k nedostatečným detailům a složitosti Bertalmiho metody jsme jeho metodu neprogramovali, ale snažili jsme se porovnat metody na základě obrazů, které Bertalmio zveřejnil na svých webových stránkách. Výběr několika obrazů, na kterých Bertalmio prováděl rekonstrukci můžete vidět níže.

a) **b)** **c)**

Obrázek 74: Bertalmiho testovací obrazy.

Hlavní porovnání rekonstrukce RBF metodou a Bertalmiho metodou musí být provedeno vizuálně a pro obraz 74a můžeme provést i porovnání metodami uvedenými v kapitole 3.5. K obrazu 74a se nám totiž podařilo získat originální nepoškozený obraz. Musíme však poznamenat, že ve výsledné chybě se může projevit i chyba způsobená převodem do adekvátního obrazového formátu pro porovnání neboť originál i poškozený obraz se nám podařilo získat od autora v rozdílných formátech.

Obrázek 75: Bertlamiho rekonstrukce obrazu s kočárem (vlevo) a naše rekonstrukce (vpravo) nejlepší metodou MAX.

Porovnání originálního obrazu vůči rekonstrukci provedené Bertalmiem a vůči RBF metodě jsou uvedené v následující tabulce. Metoda MAX dosáhla lepšího výsledku než Bertalmiho metoda. Chyba rekonstrukce je o poznaní menší.

	kanál	S	S^2	PSNR [dB]
Bertalmio	**R**	6,37	122,68	27,24
RBF (MAX)	**R**	1,80	59,78	30,36
Bertalmio	**G**	4,99	91,61	28,51
RBF (MAX)	**G**	1,69	52,13	30,96
Bertalmio	**B**	6,97	127,73	27,07
RBF (MAX)	**B**	1,70	53,30	30,86
Bertalmio	**I**	4,46	85,96	28,79
RBF (MAX)	**I**	1,68	52,75	30,91

Tabulka 33: Porovnání rekonstrukce obrazu 74a.

Obrázek 76: Bertalmiho rekonstrukce skokana na laně (vlevo), naše rekonstrukce metodou MAX (uprostřed) a detaily na rekonstruovanou část obrazu (vpravo – originál, Bertalmio, MAX).

Obrázek 77: Bertalmiho rekonstrukce rytířů (vlevo) a naše rekonstrukce (vpravo).

Z uvedených výsledků rekonstrukce je vidět, že metoda rekonstrukce pomoci RBF metody zvládá rekonstrukci poškozených obrazů velmi dobře a výsledky jsou s Bertalmiho metodou srovnatelné. Na obraz byla použita maska, kterou Bertalmio zveřejnil. K obrazům Obrázek 76, Obrázek 77 se nám nepodařilo získat nepoškozený originál a nebylo tedy možné provést stejné srovnání jako pro Obrázek 75.

Obraz	S	S^2	PSNR	$\phi(r)$
75a - R	MAX	MAX	MAX	1
75a - G	OI	MAX	MAX	1
75a - B	OI	OI	OI	1

75a - I	MAX	MAX	MAX	1

Tabulka 34: Porovnání rekonstrukce pomocí RBF s rekonstrukcí Bertalmiho metodou na obrazu 76a.

Obrázek 76 je velmi důležitý a ukazuje slabinu RBF metody, pokud je použito okno s konstantní velikostí k-okolí. V takovém případě, pokud je prováděna rekonstrukce konstantně barevné plochy, dochází k narůstání chyby a po rekonstrukci je patrné, kde byla rekonstrukce prováděna (Obrázek 76 , poslední obraz vpravo). Tento problém by se dal vyřešit dynamickou změnou velikosti k-okolí tak, aby k-okolí přesahovalo do známých hodnot v obaze přes poškozenou část.

Protože porovnání chyby dvou barevných obrazů je obecně velký problém, tak jsou v tabulce uvedeny pouze metody, které byly v daném výpočtu nejlepší. Jsou uvedeny výsledky pro všechny složky barevného obrazu včetně intenzity.

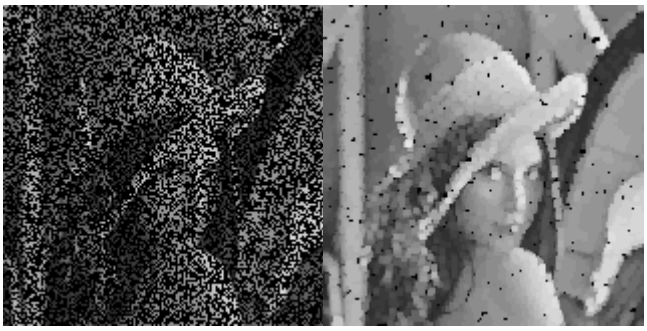
Bertalmio uvádí, že všechny výpočty proběhli do 7 minut v následující tabulce jsme shrnuli výsledky rekonstrukce a uvedli i výpočetní časy na srovnatelném stroji jež použil Bertalmio.

Obraz	R [s]	G [s]	B [s]	Celkem [s]
74a (min)	20,45	20,11	20,02	60,58
74a (max)	80,75	81,84	81,41	244
74b (min)	1,72	1,69	1,70	5,11
74b (max)	41,47	33,84	38,40	113,71
74c (min)	1,65	1,62	1,68	1,62
74c (max)	29,88	29,80	29,83	30,25

Tabulka 35: Časová náročnost rekonstrukce.

Co se týče kvality rekonstrukce a doby výpočtu, tak vychází RBF metoda s použitím TPS bázové funkce lépe než Bertalmiho metoda. Samozřejmě, že by bylo lepší udělat podrobnější analýzu v kooperaci s panem Bertalmiem.

Další oblastí rekonstrukce je oprava obrazu poškozených šumem. S takovým poškozením se můžeme setkat na starších fotografiích nebo při poškození CCD prvku digitálního fotoaparátu. Základní metodou v této oblasti je použití nějakého filtru nebo FFT filtrace. Jak takto rekonstruovaný obraz s použitím různých filtrů můžete vidět na následujících obrázcích.



a) **b)**

Obrázek 78: Filtrace poškozeného obrazu: a) poškozený obraz, b) maximální propust,

Filtry většinou dokážou odstranit jednoduchý šum určité frekvence, ale s větším poškozením si neporadí. I velmi poškozené obrazy () dokáže naše metoda opravit.

3.10.2 CSRBF

Jako poslední jsme zvolili porovnání s metodou uvedenou v [Kojekine04], která provádí rekonstrukci RBF metodou a využívá k tomu compactly-supported bázové funkce. Metoda velmi dobře zvládá rekonstruovat různé druhy poškození, velmi dobře využívá výhod lokality CSRBF a požívá metody urychlující řešení lineárního systému. Oproti tomu naše metoda využívá globální bázovou funkci TPS. Výpočet je prováděn v putujícím okně a pro řešení lineárního systému využíváme základní metodu, LU faktorizaci.

Porovnání kvality rekonstrukce obou metod bude opět poze na vizuální úrovni neboť nemáme k dispozici originální nepoškozené obrazy. Byl vybrán jeden z několika obrazů pro možnost porovnání výsledků rekonstrukce. [Karlson].

a) **b)** **c)**

Obrázek 79: Rekonstrukce obrazu s vážkou. (a) poškozený obraz, b) rekonstrukce CSRBF, c) rekonstrukce TPS metodou MAX)

Výhod našeho řešení proti řešení využívající CSRBF je několik. Jako první můžeme uvést problémy při řešení velmi poškozených obrazů šumem. CSRBF má v tomto případě problémy. Naše metoda dokáže takto poškozené obrazy opravit. Další problém CSRBF metody je nutnost nastavovat parametr α, který určuje míru vlivu CS bázové funkce na její okolí. Nastavování musí být provedeno v závislosti na poškození obrazu. V našem případě používáme globální bázovou funkci TPS, která nemá žádný volitelný parametr. Významným hlediskem je to, že jsem se bez významného urychlování metody velmi přiblížili časové náročnosti CSRBF metody.

Nevýhoda naší metody může být hlavně při rekonstrukci škrábanci poškozeného obrazu, kdy CSRBF metoda dokáže, s vhodně nastaveným parametrem α, zrekonstruovat celý obraz během jednoho průchodu.

4 Závěr

V uvedené práci jsem se zabýval použitím radiálních bázových funkcí v počítačové grafice a zpracování obrazu. Základní oblastí, kde jsem se snažil touto prací prohloubit znalosti, byla dosud neprozkoumaná oblast zpracování obrazu. Konkrétně se jedná využití radiálních bázových funkcí na rekonstrukci různě poškozených obrazů.

Definoval jsem několik různých nových metod založených na principu rekonstrukce poškozeného obrazového bodu ze znalosti hodnot jiných obrazových bodu z jeho k-okolí. Většina metod používá při rekonstrukci putující okno, ale je zde i metoda vyhledávající v obraze nejvhodnější body k rekonstrukci. Provedl jsem porovnání různých bázových funkcí a stanovil vhodnost jejich použití při rekonstrukci. Navržené metody jsem konfrontoval s jinými existujícími metodami pro rekonstrukci poškozených obrazů a taktéž s prakticky jedinou prací využívající k rekonstrukci poškozených obrazů radiálních bázových funkcí [Kojekine04].

Navržené metody poskytují výborné výsledky při rekonstrukci a zvládají opravit širokou škálu poškození oproti jiným metodám, které jsou velmi často zaměřené pouze na jeden druh poškození. V kapitole 3.2.9 je provedeno komplexní porovnání navržených metod, ze kterého vychází MAX metoda jako metoda poskytující nejlepší výsledky na celém spektru poškození. Bohužel je tato metoda nejpomalejší. Záleží tedy zda uživatel upřednostní rychlost nebo kvalitu. V porovnání s ostatními metodami dokáží nové metody překonat v kvalitě metody stávající. Při porovnávání rekonstruovaných obrazů bylo sledováno několik základních faktorů jakými jsou MSE chyba nebo PSNR (kapitola 3.5).

Uvedené metody mají samozřejmě potenciál na další vylepšení. Existuje několik oblastí rozšíření jako například využití různých bázových funkcí v jednom obraze, analýza hran v obraze a přizpůsobení rekonstrukce na hranách nebo dynamická změna velikosti okna. Těmito směry se může vydat další výzkum v této oblasti.

Všechny výpočty byly provedeny na počítači Intel Pentium M, 1.7 GHz, 1GB RAM. Pro velkou obsáhlost testů nejsou všechny detaily a výsledky uvedeny přímo v prácí, ale na přiloženém DVD.

Implementace všech uvedených metod byla provedena v jazyce C# na platformě .NET s použitím volně dostupně matematické knihovny DotNetMatrix [DotNetMatrix].

Literatura

[ARANZ] Applied Research Associates NZ Ltd: http://aranz.com/

[Barnhill77] R. Barnhill, Representation and Approximation of Surfaces, J.R. Rice, ed., *Mathematical Software III*, pp. 68-119. New York: Academic Press, 1977.

[Bastl01] Bastl, B.: "Metody moderní algebry a jejich aplikace", Thesis, University of West Bohemia, Faculty of Applied Sciences, Department of Mathematics, Czech Republic, 2001.

[Bastl03] Bastl, B.: Metody eliminace proměnných pro soustavy nelineárních algebraických rovnic a jejich aplikace v geometrickém modelování, State of the Art and Concept of Doctoral Thesis, University of West Bohemia, Faculty of Applied Sciences, Department of Mathematics, Czech Republic, 2003.

[Beatson97] R. K. Beatson and L. Greengard. A short course on fast multipole methods. In M. Ainsworth, J. Levesley, W.A. Light, and M. Marletta, editors, *Wavelets, Multilevel Methods and Elliptic PDEs*, pages 1–37. Oxford University Press, 1997.

[Beatson99] Beatson, R. K., Cherrie, J. B., Mouat, C. T.: Fast fitting of radial basis functions: Methods based on preconditioned GMRES iteration, *Advances in Computational Mathematics*, vol. 11, pp. 253-270, 1999.

[Beatson00] Beatson, R.K., Light, W.A., Billings, S.: Fast solution of the radial basis function interpolation equations: Domain decomposition methods, *SIAM J., Sci. Comput.*, Vol. 22, pp. 1717-1740, 2000.

[Bertalmio00] Bertalmio, M., Sapiro, G.: Image Inpainting, Proceedings of SIGGRAPH'00, Computer Graphics, New Orleans, 23-28 July 2000, pp.417-424.

[Burt88] P.J. Burt, Moment Images, Polynomial Fit Filters, and the Problem of Surface Interpolation, *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 144-152, 1988.

[Carr97] Carr, J. C., Fright, W. R., Beatson, R. K.: Surface interpolation with radial basis functions for medical imaging, *IEEE Trans. Medical Imaging*, vol. 16, pp. 96-107, February 1997.

[Carr01] Carr, J. C., Beatson, R. K., Cherrie, J. B., Mitchell, T. J., Fright, W. R., McCallum, B. C., Evans, T. R.: Reconstruction and representation of 3D objects with radial basis functions, *Computer Graphics (SIGGRAPH 2001 proceedings)*, pp. 67-76, August 2001.

[Clough65] R. Clough and J. Tocher, Finite Element Stiffness Matrices for Analysis of Plates in Bending, *Proc. Conf. Matrix Methods in Structural Mechanics*, pp. 515-545, 1965.

[DotNetMatrix] DotNetMatrix, port of a public domain Java matrix library, JAMA [online], http://math.nist.gov/javanumerics/jama

[Duchon77] Duchon, J.: Splines minimizing rotation-invariant semi-norms in Sobolev space, Constructive Theory of Functions of Several Variables, *Springer Lecture Notes in Math*, vol. 21. pp. 85-100, 1977.

[Fornberg02] Fornberg B., Driscoll T.A., Wright G., Charles R.: Observations on the behavior of radial basis functions near boundaries, *Comput. Math. Appl.*, vol. 43. pp. 473-490, 2002.

[Fornberg04] Fornberg B., Wright G.: Stable Computation of Moltiquadric Interpolants for All Values of the Shape Parameter, *Comput. Math. Appl.*, 2004.

[Franc02] Franc, M.: Methods for Polygonal Mesh Simplification. State of the Art and Concept of Doctoral Thesis, Technical Report No. DCSE/TR-2002-01, University of West Bohemia, Pilsen, Czech Republic, January 2002.

[Franke79] Franke, R.: "A critical comparison of some methods for interpolation of scattered data", TR NPS-53-79-003, Naval Postgraduate School, 1979.
[Franke82] Franke, R.: Scattered Data Interpolation: Tests of Some Method, *Mathematics of Computation*, Vol.38, No. 157, 1982, pp. 181-200.
[Glassner95] A. Glassner, Principles of Digital Image Synthesis. San Francisco, Calif.: *Morgan Kaufmann*, 1995.
[Goshtashby87] A. Goshtasby, "Piecewise Cubic Mapping Functions for Image Registration," *Pattern Recognition*, vol. 20, no. 5, pp. 525-533, 1987.
[Greengard87] Greengard, L., Rokhlin, V.: "A fast algorithm for particle simulations", *J. Comput. Phys.*, 73(2):325--348, 1987.
[Hardy71] Hardy, L.R.: Multiquadric equations of topography and other irregular surfaces, *J. Geophy. Res.*, vol. 76. pp. 1905-1915, 1971.
[Hardy75] Hardy, L.R., Gopfert, W.M.: Least squares prediction of gravity anomalies, geoidal undulations, and deflection of the vertical with multiquadric harmonic functions, *Geophys. Res.*, vol. 10., pp. 423-426, 1975.
[Hardy77] Hardy, L.R.: Least square prediction. *Photogramm. Engng Rem. Sens.* 43, pp. 475-492, 1977.
[Hardy90] Hardy, L.R.: Theory and Applications of the Multiquadric-Biharmonic Method: 20 Years of Discovery, *Computers Math. Applic.*, Vol 19, pp. 163-208, 1990.
[Hirani96] Hirani, A., Totsuka, T.: Combining Frequency and spatial domain information for fast interactive image noise removal, *Computer Graphics*, pp. 269-276, *SIGGRAPH 96*, 1996.
[Jennings66] Jennings, A.: A Compact Storage Scheme for The Solution of Symmetric Linear Simultaneous Equations, *Comput. Journal*, Vol. 9, pp. 281-285, 1966.
[Karlson] http://www.karlson.ru/csrbf/index.php?id=5
[Kojekine04] Kojekine, N., Savchenko, V.: Using CSRBF for surface retouching, *IPSJ (Information Processing Society of Japan) Journal*, 2004 (in japanese).
[Kozhekin03] Kozhekin, N., Savchenko, V., Senin, M., Hagiwara, I.: An approach to surface retouching and mesh smoothing, *The Visual Computer*, vol. 19, pp. 549-564, 2003.
[Laga02] Laga, H., Piperakis, R., Takahashi, H., Nakajima, M.: "3D Object Reconstruction from Scanned Data using Radial Basis Functions and Volumetric Processing", *Nicograph 2002 National Conference*, Nagoya, Japan. October 2002, pp133-138.
[Lawson77] C. Lawson, "Software for C^1 Surface Interpolation," *Mathematical Software III*, J.R. Rice, ed., pp. 161-194. New York: Academic Press, 1977.
[Lee97] Lee, S.,Wolberg, G., Sung, Y.S.: Scattered Data Interpolation with Multilevel B-Splines, *IEEE Transaction of Visual Computational Graphics 3*, Vol. 3, pp. 228-244., 1997.
[Madych92a] Madych, W.R.: "Miscellaneous error bounds for multiquadric and related interpolants", *Comput. Math. Appl.*, pp. 121-138, 1992.
[Madych92b] Madych, W.R., Nelson, S.A.: "Bounds on multivariate polynomials ad exponential error estimates for multiquadric interpolation", *J. Approx. Theory*, Vol. 70, pp. 94-114, 1992.
[Micchelli86] Micchelli, C.A.: "Interpolation of scattered data: distance matrice and conditionally positive definite functions", *Constr. Approx.*, vol. 2., pp. 11-22, 1986.
[Mika96] Míka, S.: "Numerické metody Lineární algebra", ZČU Plzeň, 1996. (Czech language)
[Morse01] Morse, B., Yoo, T. S., Rheingans, P., Chen, D. T., Subramanian, K.R.: "Interpolating implicit surfaces from scattered surface data using compactly

supported radial basis functions", in *Proceedings of the Shape Modeling conference, Genova*, Italy, 89-98, May 2001.

[Nitzberg93] Nitzberg, M., Mumford, D., Shiota, T.: Filtering, Segmentation and Depth, *Springer-Verlag*, Berlin, 1993.

[Othake04a] Othake Y., Belyaev A., Seidel H.P.: 3D Scattered Data Approximation with Adaptive Compactly Supported Radial Basis Functions, *SMI'2004*, 2004.

[Othake04b] Othake Y., Belyaev A., Seidel H.P.: Multi-Scale and Adaptive CS-RBFs for Shape Reconstruction from Cloud of Points, *Advances in Multiresolution for Geometric Modelling*, N.Dodgson, M.S.Floater, M.Sabin (Eds.), Springer, 2004.

[Platte05] Platte, B.R, Driscool, A.T.: Polynomials and Potential Theory for Gaussian Radial Basis Function Interpolation, *To appear in SIAM J. Numer. Anal.*

[Powell92] Powell, M.J.D.: The theory of radial basis function approximation in 1990, in Advances in Numerical Analysis, Vol. II: Wavelets, subdivision Algorithms and Radial Functions, W.Light, ed., *Oxford University Press*, Oxford, UK, pp. 105-210, 1992.

[Press97] Press, W.H., Teukolsky, S.A., Vetterling, T., Flannery, B.P.: Numerical Recipes in C, *Cambridge University Press*, 1997.

[Rektorys95] Rektorys, K.: Přehled užité matematiky I,II, *Nakladatelství Prométheus*, Praha, 1995.

[Savchenko02] Savchenko V, Kojekine N, Unno H.: A practical image retouching method. In: *Proceedings of the international symposium Cyber Worlds'02: theory and practice*, Tokyo, 6–8 November 2002, pp 480–487, 2002.

[Shepard68] Shepard, D.: A Two Dimensional Interpolation Function for Irregularly Spaced Data, *Proc. ACM 23rd Nat'l Conf.*, pp. 517-524, 1968.

[Schaback94] Schaback, R.: Error estimates and condition numbers for radial basis function interpolants, *Adv. Comput. Math.*, Vol. 3, pp. 251-264, 1994.

[Schagen79] Schagen, I.P.: Interpolation in Two Dimension – A New Technique, *J. Inst. Maths Applics*, vol. 23, pp. 53-59, 1979.

[Schoenberg38] Schoenberg, I.J.: Metric Spaces and Completely Monotone Functions, *The Annals of Mathematics*, 2nd Ser., Vol.39, No.4, pp.811-841, 1938.

[Sprott04] Sprott, J.C.: A method for approximation missing data in spatial patterns, *Computer & Graphics*, vol. 28, pp. 113-117, 2004.

[Tobor04] Tobor I., Reuter P., Schlick Ch.: Multiresolution Reconstruction of Implicit Surface with Attributes from Large Unorganized Point Set, in *Proceedings of Shape Modeling International* (SMI 2004), 2004.

[Turk02] Turk, G., O'Brien, J.: Modelling with implicit Surface that interpolate, *ASM Transactions on Graphics*, Vol. 21, No. 4, October 2002.

[Turk99] Turk, G., O'Brien, J.: Shape Transformation Using Variational Implicit Functions, *SIGGRAPH 99*, August 1999, pp. 335-342, 1999.

[Uhlir01] Uhlíř, K., Skala, V.: Interaktivní system pro generovaní implicitních funkcí a jejich modelování, Thesis, University of West Bohemia, Faculty of Applied Sciences, Department of Computer Science and Engineering, Czech Republic, 2001. (Czech language)

[Uhlir02] Uhlir, K., Skala, V.: Kompilovaný HyperFun, Technical Report DCSE/TR-2002-07, University of West Bohemia, Czech Republic, 2002. (Czech language)

[Uhlir03] Uhlir, K., Skala,V.: The Implicit Function Modelling System - Comparison of C++ and C# Solutions, *C# and .NET Technologies'2003*, University of West Bohemia, Czech Republic, ISBN 80-903100-3-6, 2003.

[Wendland05] Wendland, H.: Computational aspects of radial basis function approximation, in K. Jetter et al. (eds.) *Topics in Multivariate Approximation and Interpolation* , Elsevier B.V., pp. 231-256, 2005.

[Wendland95] Wendland, H.: Piecewise polynomial, positive definite and compactly supported radial basis functions of minimal degree, *Advances in Computational Mathematics 4* (1995), pp. 389-396, 1995.

[Wendland98] Wendland, H.: Error estimates for interpolation by compactly supported radial basis functions of minimal degree, *Journal of Approximation Theory 93*, 258-272, 1998.

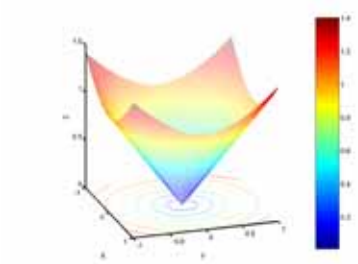
[Wright03] Wright, G.B., Radial Basis Function Interpolation: Numerical and Analytical Developments, University of Colorado, Thesis, 2003.

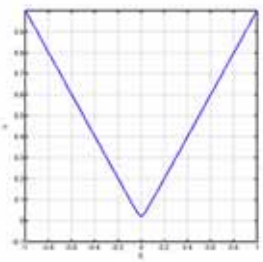
[Wu95] Wu Z.: Compactly supported positive definite radial functions, *Advences in Computational Mathematics 4* (1995), 283-292, 1995.

[Yngve02] Yngve, G., Turk, G.: Robust Creation of Implicit Surfaces from Polygonal Meshes, *IEEE Transactions on Vizualization and Computer Graphics*, Vol. 8, No. 4, pp. 346-359, October-December 2002.

[Yoo01] Yoo, T. S., Morse, B., Subramanian, K., Rheingans, P., Ackerman, M. J.: Anatomic modeling from unstructured samples using variational implicit surfaces, *Medicine Meets Virtual Reality*, 2001.

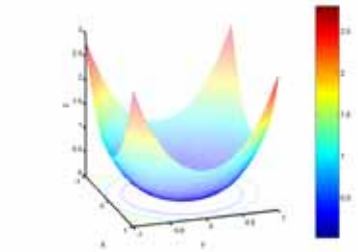
PŘÍLOHA A – Bázové funkce





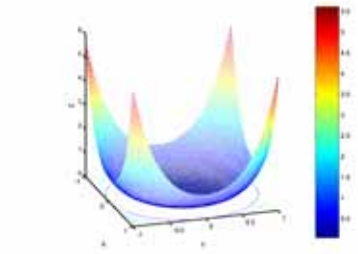
Lineární

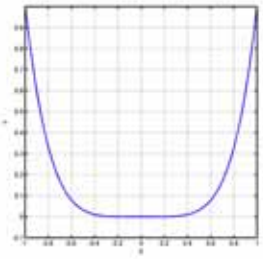
Funkce: $\phi(r) = r$



Kubická

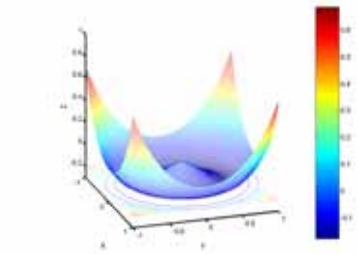
Funkce: $\phi(r) = r^3$

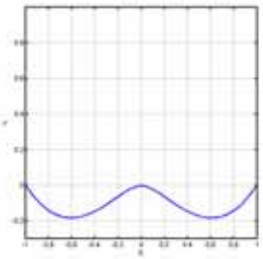




Quantic

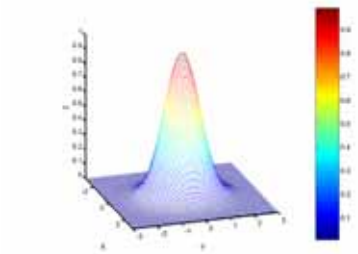
Funkce: $\phi(r) = r^5$

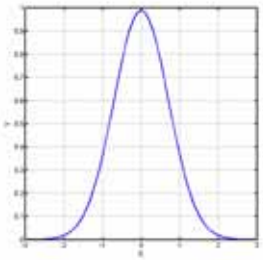




Thin-plate spline (TPS)

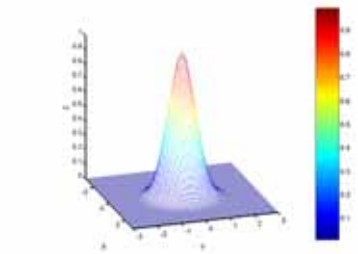
Funkce:
$\phi(r) = r^2 \log r$

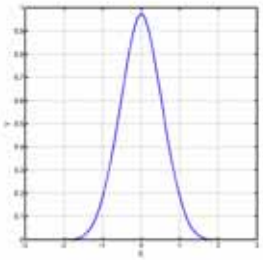




Gaussova (GRBF)

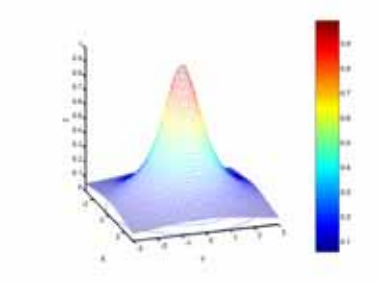
Funkce: $\phi(r) = e^{-(\varepsilon r)^2}$
$\varepsilon = 1$

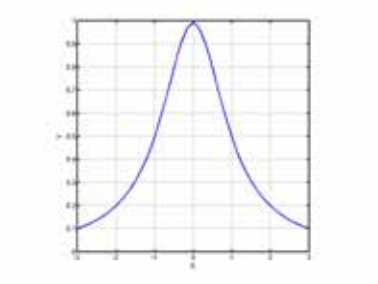




Compactly supported (CSRBF), C^2

Funkce:
$\phi(r) = (1-r)_+^4 (4r+1)$



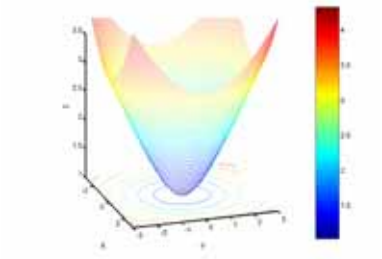


Inverzní kvadratická (IQ)

Funkce:

$$\phi(r) = \frac{1}{1+(\varepsilon r)^2}$$

$$\varepsilon = 1$$

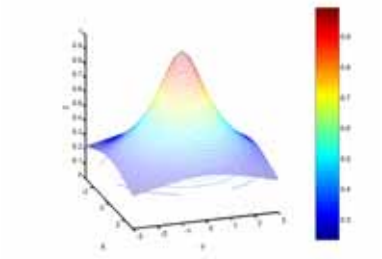


Multiquadric (MQ)

Funkce:

$$\phi(r) = \sqrt{1+(\varepsilon r)^2}$$

$$\varepsilon = 1$$

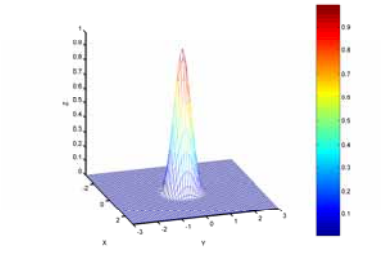


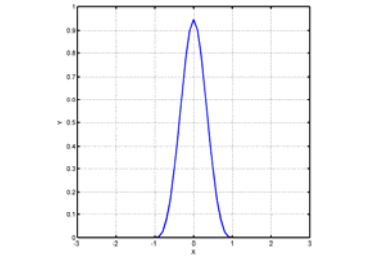
Inversní multiquadric (IMQ)

Funkce:

$$\phi(r) = \frac{1}{\sqrt{1+(\varepsilon r)^2}}$$

$$\varepsilon = 1$$

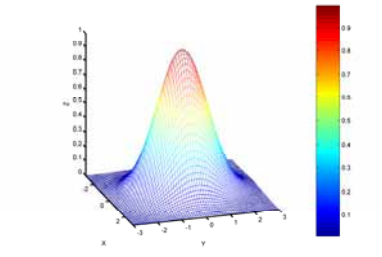


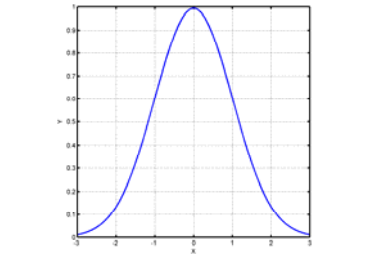


Compactly supported (CSRBF), C^2

Funkce:

$$\phi(r) = (1-r)_+^3 (3r+1)$$

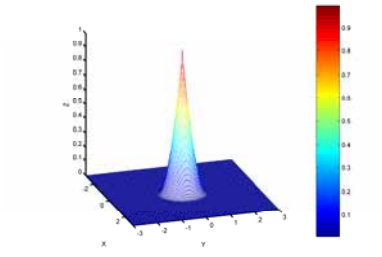


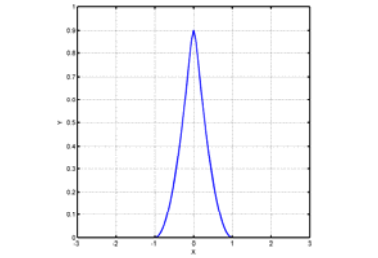


Gaussova (GRBF)
[Schagen79]

Funkce: $\phi(r) = e^{-\frac{r^2}{2\varepsilon^2}}$

$$\varepsilon = 1$$



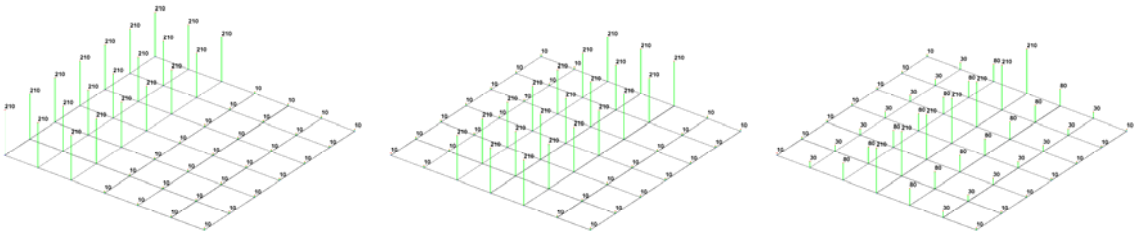


Compactly supported (CSRBF), C^0

Funkce: $\phi(r) = (1-r)_+^2$

PŘÍLOHA B – Interpolace různými bázovými funkcemi

Příklad:
Otestujme interpolaci specifických případů nastávajících v obrazech pomocí různých bázových funkcí.



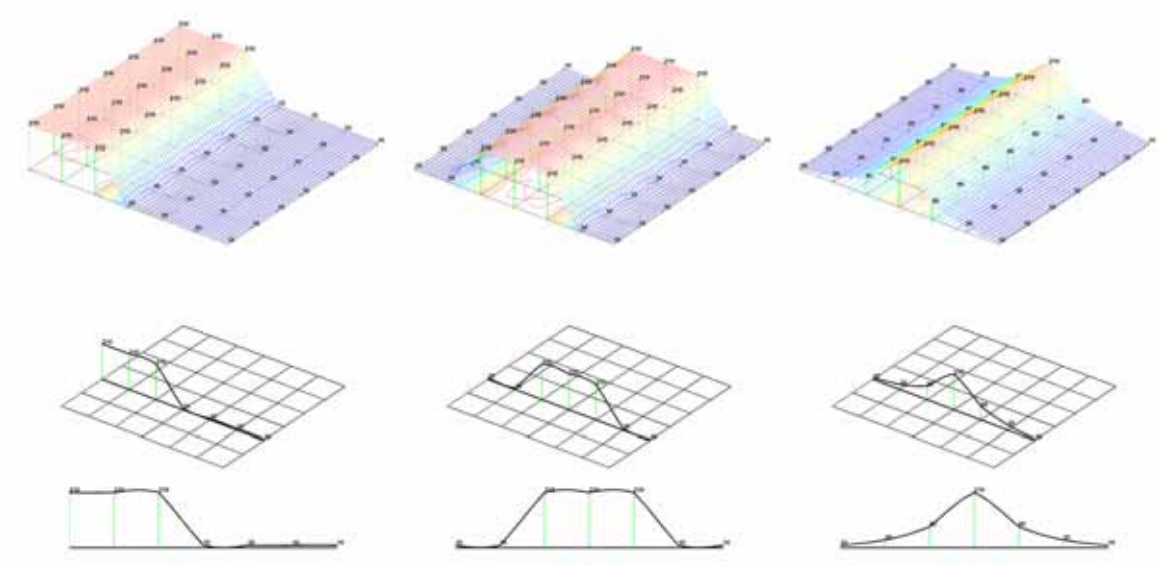
a) **b)** **c)**

Obrázek 1: Základní případy změny hodnoty jasu v obraze: a) skokově v jednom směru, b) skokově v obou směrech (např. černé okraje objektu), c) pozvolný přechod.

Základní případy, ke kterým dochází na hranách jsou přechody mezi intenzitami. Většinou dochází k několika základním případů, jež jsou zobrazeny na Obrázek 1. Na těchto případech si ukážeme jak je zvládají jednotlivé funkce interpolovat.

Lineární bázová funkce $\phi(r) = r$

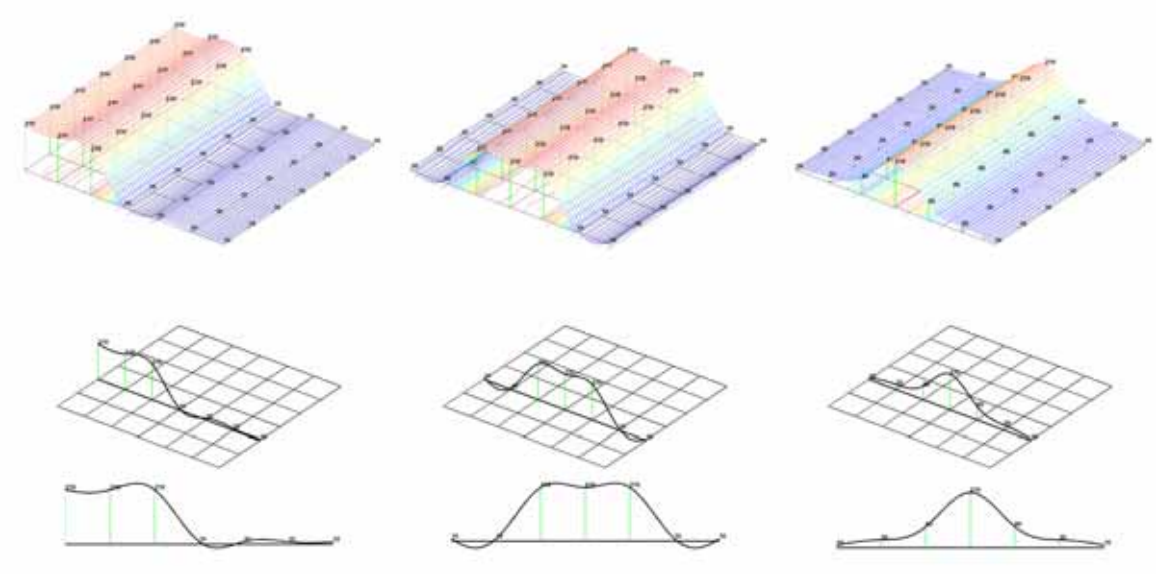
Tato funkce je vlastně určena pouze vzdáleností mezi body.



Obrázek 2: Interpolace lineární bázovou funkcí.

Kubická bázová funkce $\phi(r) = r^3$

Kubická bázová funkce se stejně jako TPS velice rychle přizpůsobuje změnám a interpolace taktéž vypadá velice kvalitně. Pouze si můžeme všimnout (Obrázek 3b dole), že je o trochu pomalejší něž TPS bázová funkce neboť přechod přes osu je něco větší než u TPS bázové funkce.

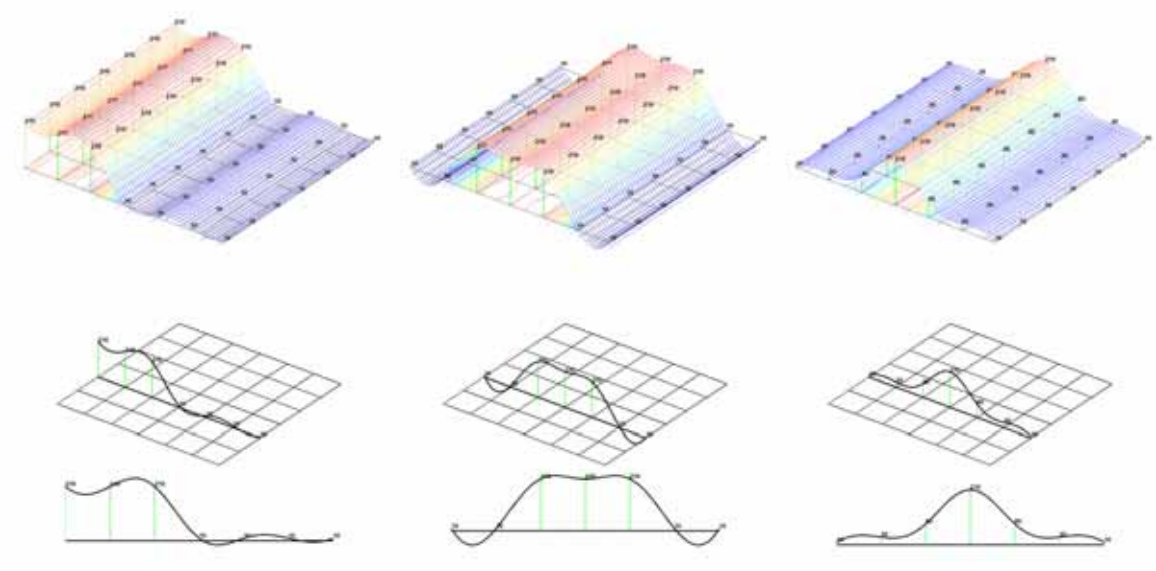


a) **b)** **c)**

Obrázek 3: Interpolace jasových hodnot kubickou bázovou funkcí.

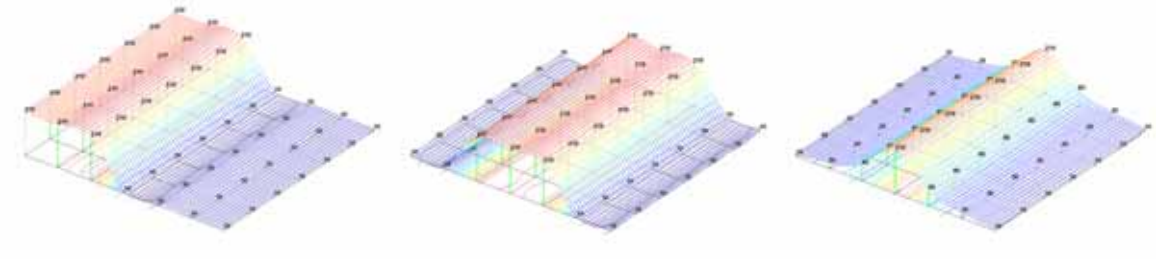
Quantic $\phi(r) = r^5$

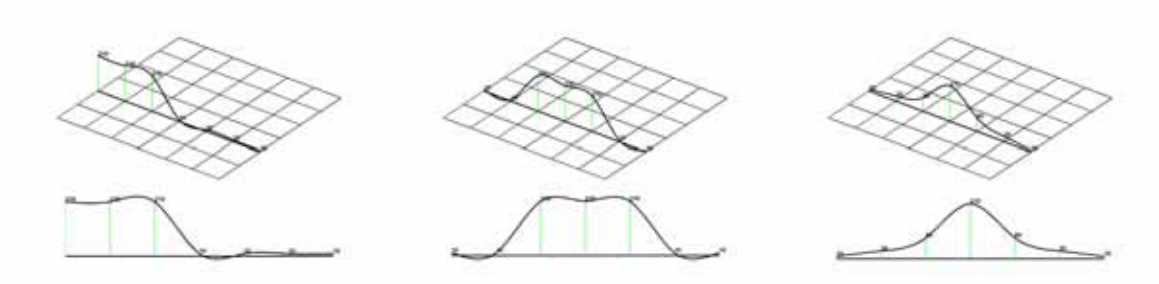
Stejně jako se kubická RBF vztahuje ke kubické spline funkci, tak se i quantic RBF interpolace vztahuje ke quantic spline funkci. [Fornberg02].



Obrázek 4: Interpolace *quantic* bázovou funkcí s parametrem.

TPS bázová funkce $\phi(r) = r^2 \log r$





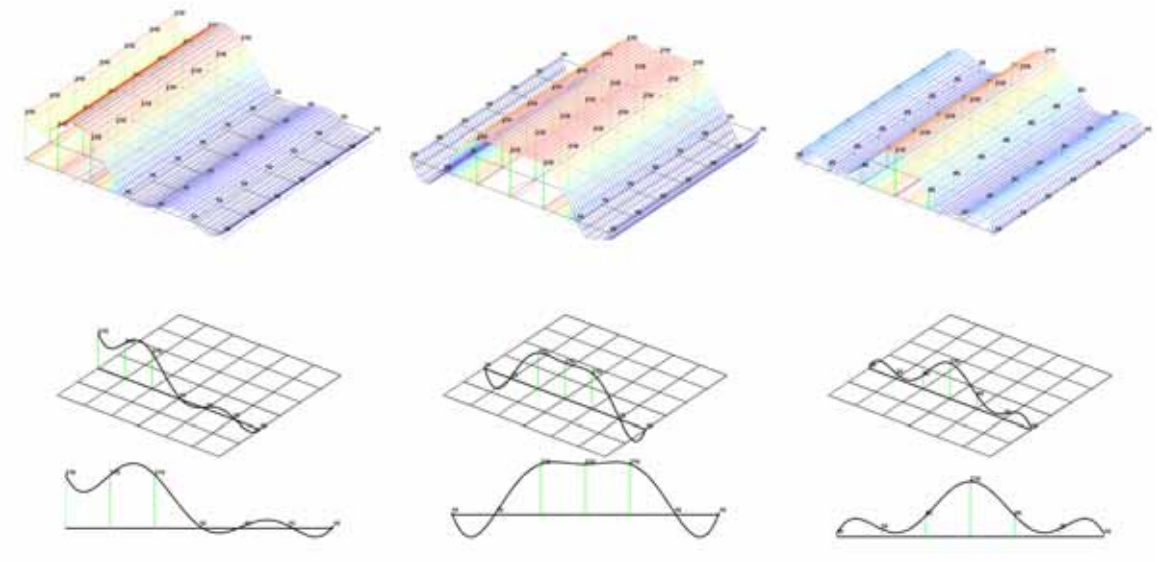
a) **b)** **c)**

Obrázek 5: Interpolace dat TPS bázovou funkcí.

TPS bázová funkce nemá žádný volitelný parametr. Interpolace TPS bázovou funkcí vypadá velice dobře. TPS se rychle přizpůsobuje změnám.

GRBF bázová funkce $\phi(r) = e^{-(\varepsilon r)^2}$

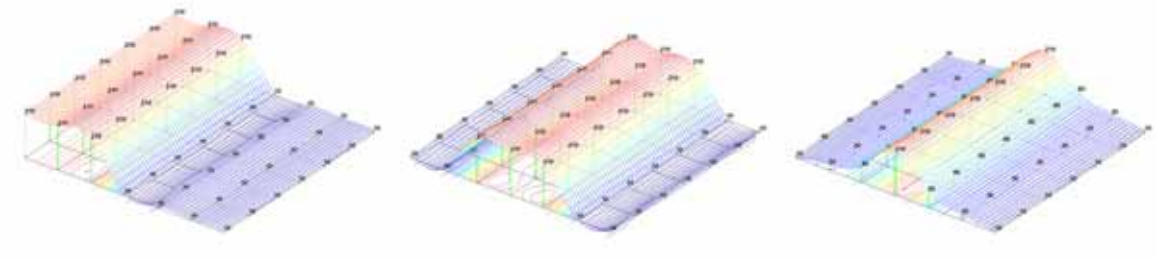
GRBF bázová funkce má volitelný parametr ε, jež ovlivňuje tvar funkce.

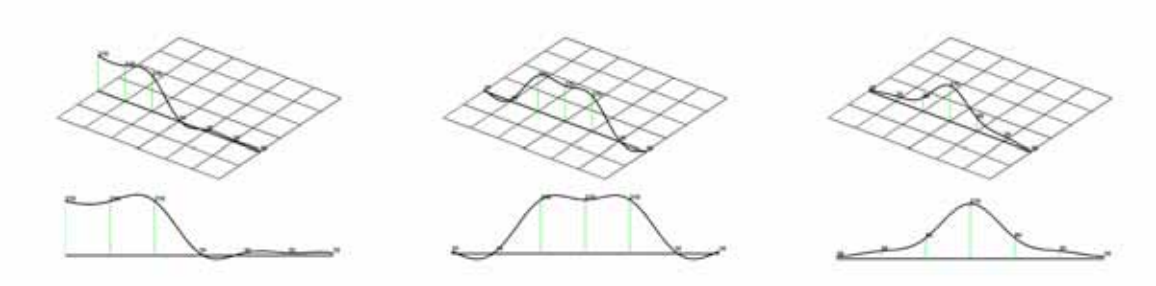


a) **b)** **c)**

Obrázek 6: Interpolace GRBF bázovou funkcí s parametrem ε = 0.1.

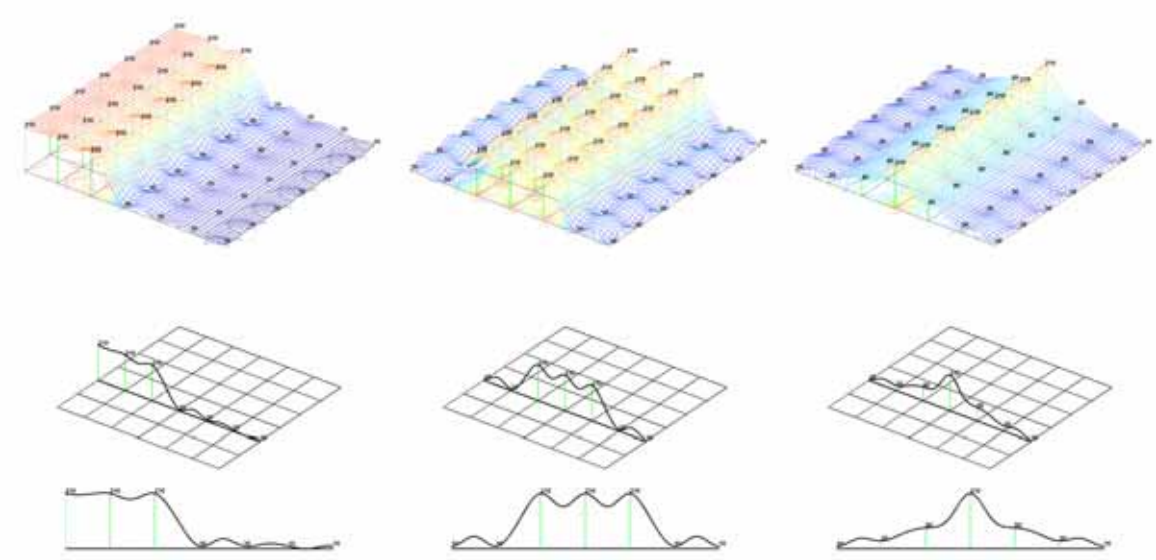
U GRBF bázové funkce (Obrázek 6) si můžeme všimnout velkého zákmitu přes osu. Funkce s tímto parametrem se špatně přizpůsobuje změnám. Vhodnější je parametr ε = 1.0 (Obrázek 7). Naopak při příliš velké hodnotě parametru ε (Obrázek 8) se funkce přizpůsobuje příliš rychle a křivka dostává mezi body tvar bázové funkce. Jak již bylo řečeno, vlastnosti interpolace se mohou měnit i se změnou rozložení vstupních dat.





a) **b)** **c)**

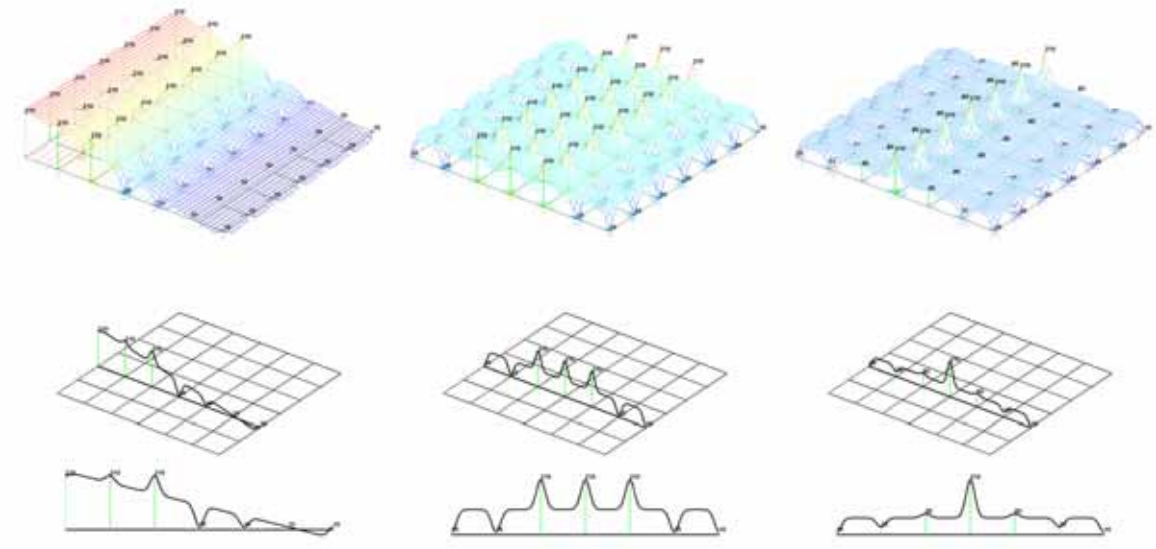
Obrázek 7: Interpolace GRBF bázovou funkcí s parametrem ε = 1.0.



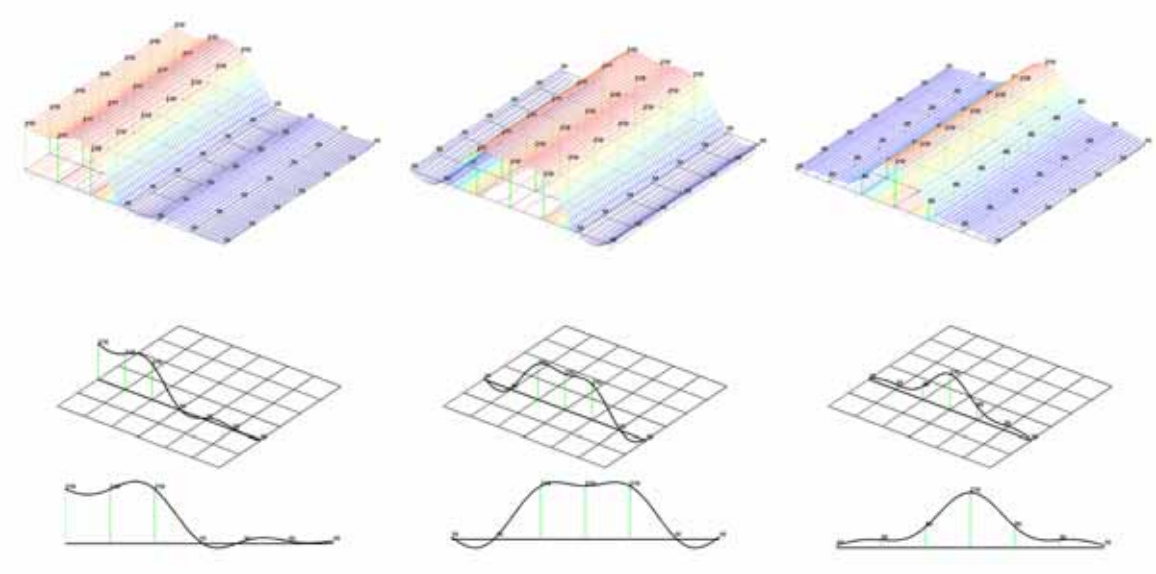
Obrázek 8: Interpolace GRBF bázovou funkcí s parametrem ε = 5.0.

GRBF bázová funkce $\phi(r) = e^{\frac{-r^2}{2\varepsilon^2}}$ [Schagen79]

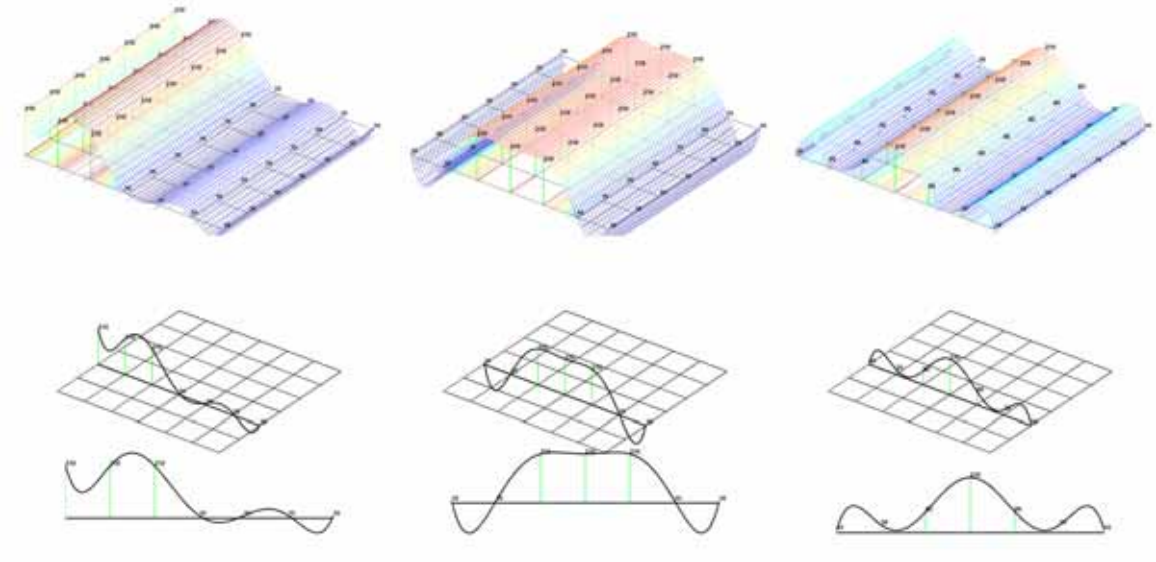
Následující obrazy ukazují interpolaci dat Schagenovou GRBF funkcí. Schagen definoval podmínky určení parametru ε.



Obrázek 9: Interpolace Schagenovou GRBF bázovou funkcí s parametrem ε = 0.1.



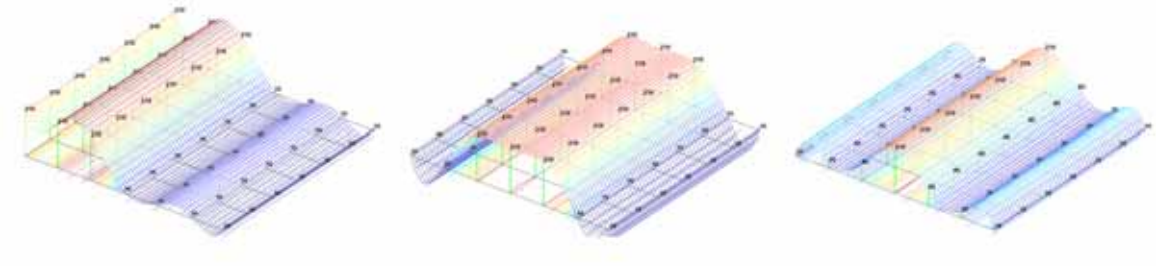
Obrázek 10: Interpolace Schagenovou GRBF bázovou funkcí s parametrem ε = 1.0.

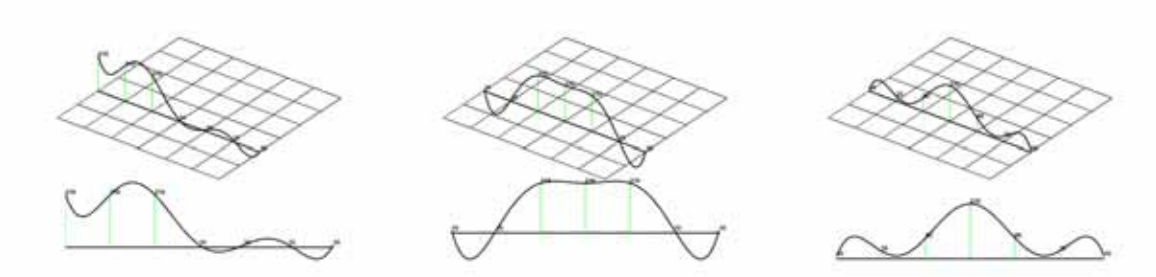


Obrázek 11: Interpolace Schagenovou GRBF bázovou funkcí s parametrem ε = 5.0.

Z uvedených třech voleb parametru ε je nejlepší volba ε = 1.0. Vyplívá to také ze způsobu stanovení parametru uvedeného v [Schagen79].

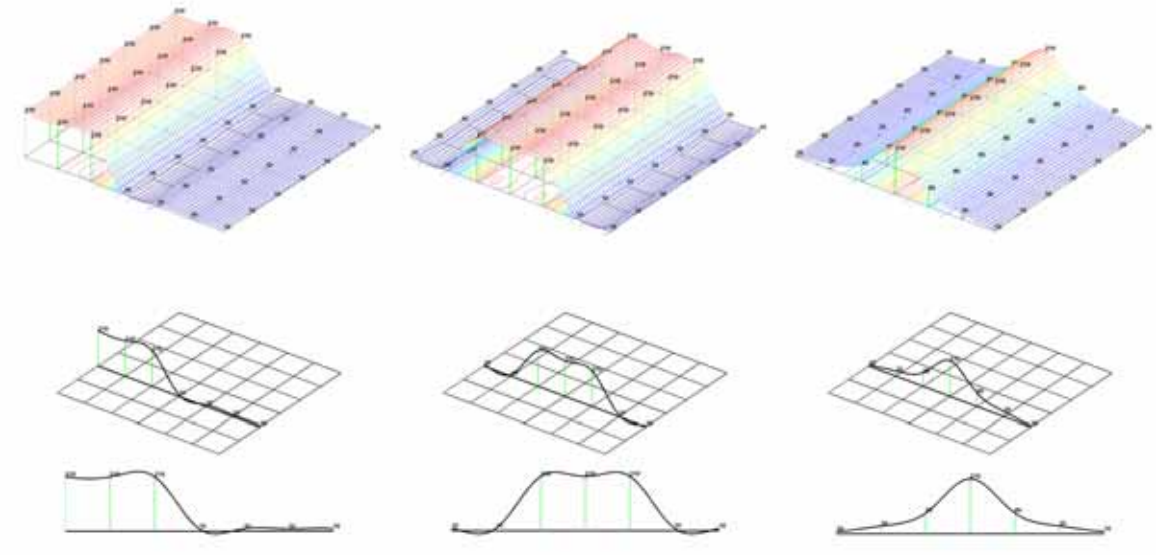
Inverzní kvadratická (IQ) $\phi(r) = \dfrac{1}{1+(\varepsilon r)^2}$



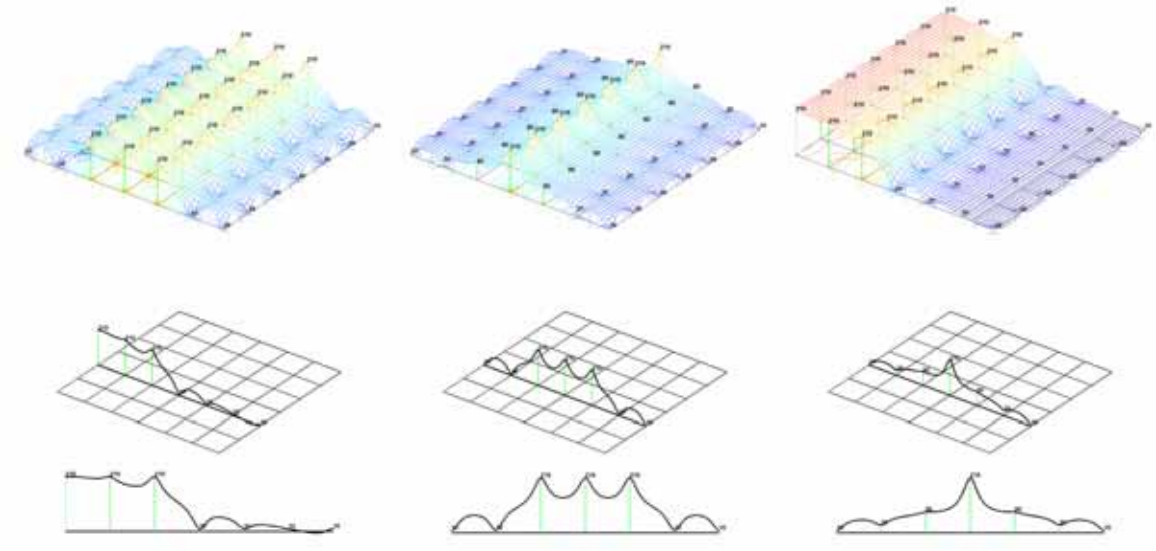


Obrázek 12: Interpolace IQ bázovou funkcí s parametrem ε = 0.1.

Je to jedna z prvních bázových funkcí použitých pro RBF metodu. Parametr ε dokáže velice změnit chování bázové funkce. Všimněme si jak se funkce s parametrem ε = 1.0 velmi dobře přizpůsobuje změnám (Obrázek 13).



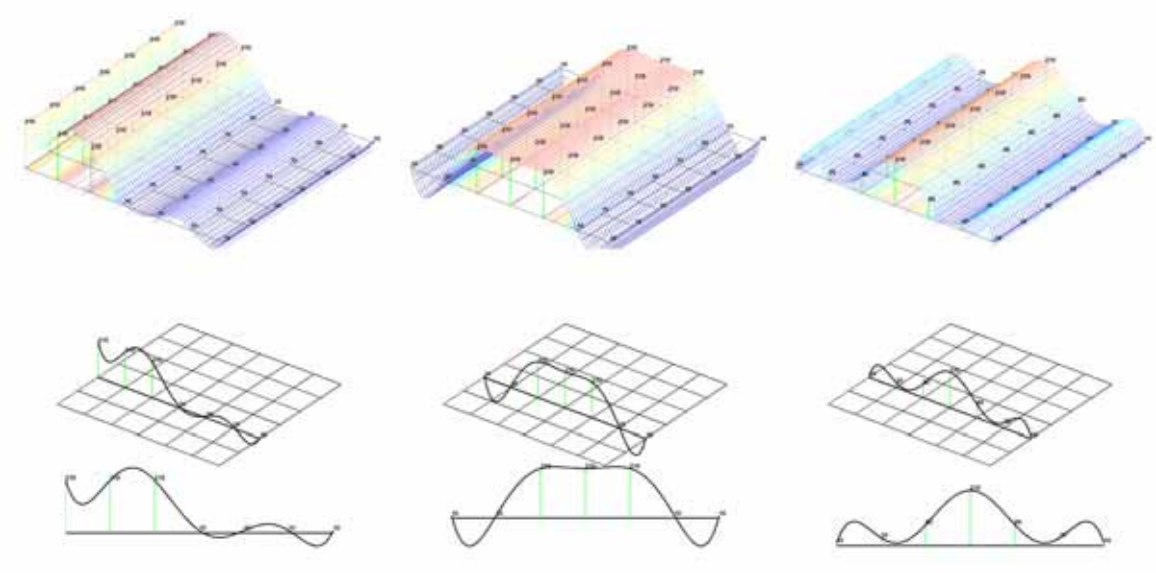
Obrázek 13: Interpolace IQ bázovou funkcí s parametrem ε = 1.0.



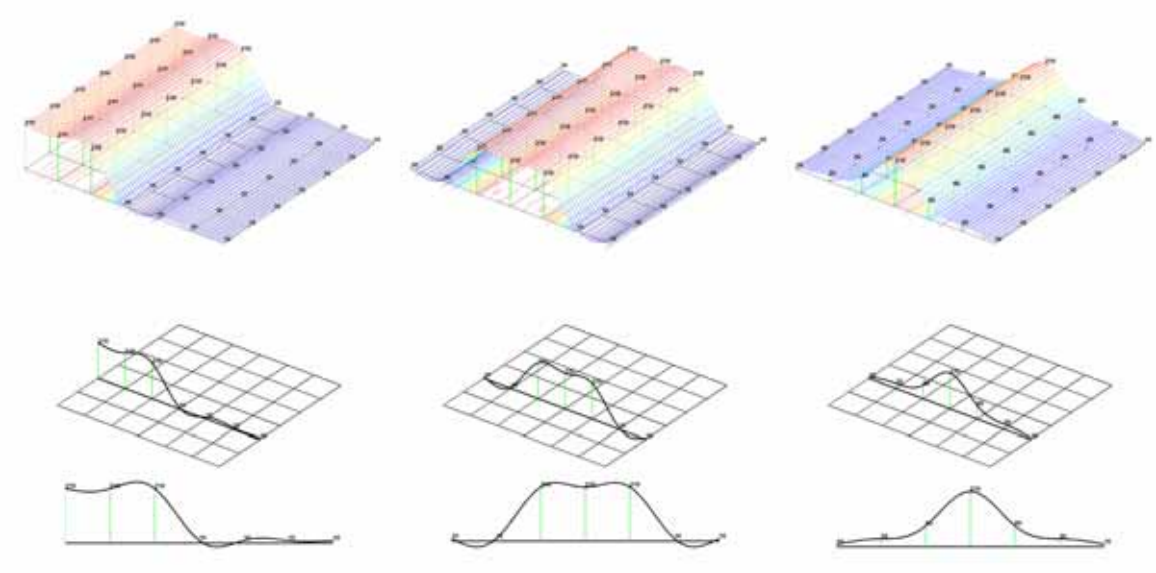
Obrázek 14: Interpolace IQ bázovou funkcí s parametrem ε = 5.0.

Multiquadric (MQ) $\phi(r) = \sqrt{1 + (\varepsilon r)^2}$

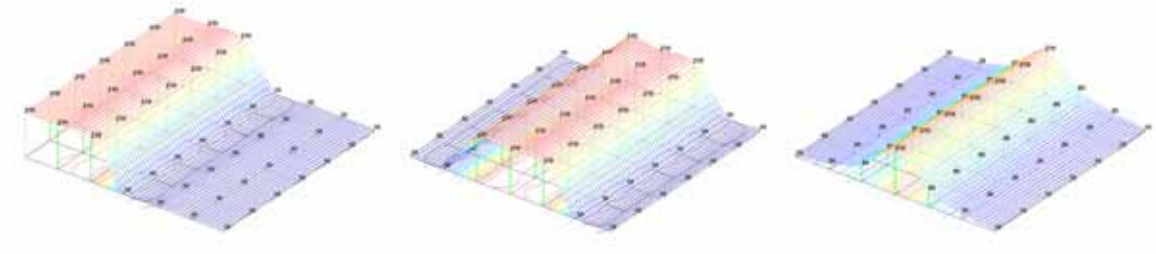
Toto je první bázová funkce pro RBF metodu používaná hlavně pro interpolaci geografických informací [Hardy71, Hardy75, Hardy90].

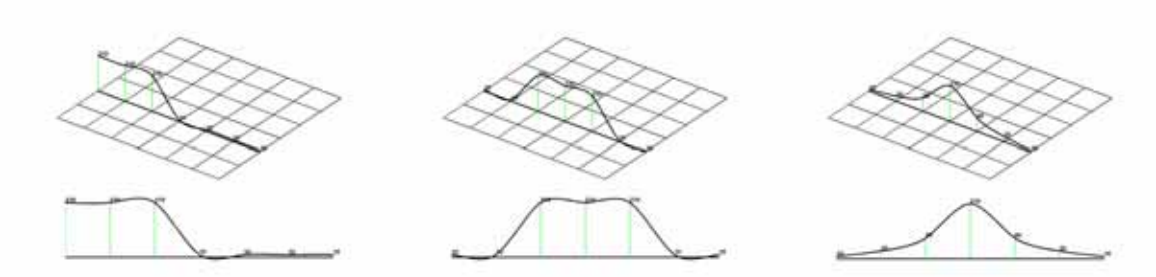


Obrázek 15: Interpolace MQ bázovou funkcí s parametrem ε = 0.1.



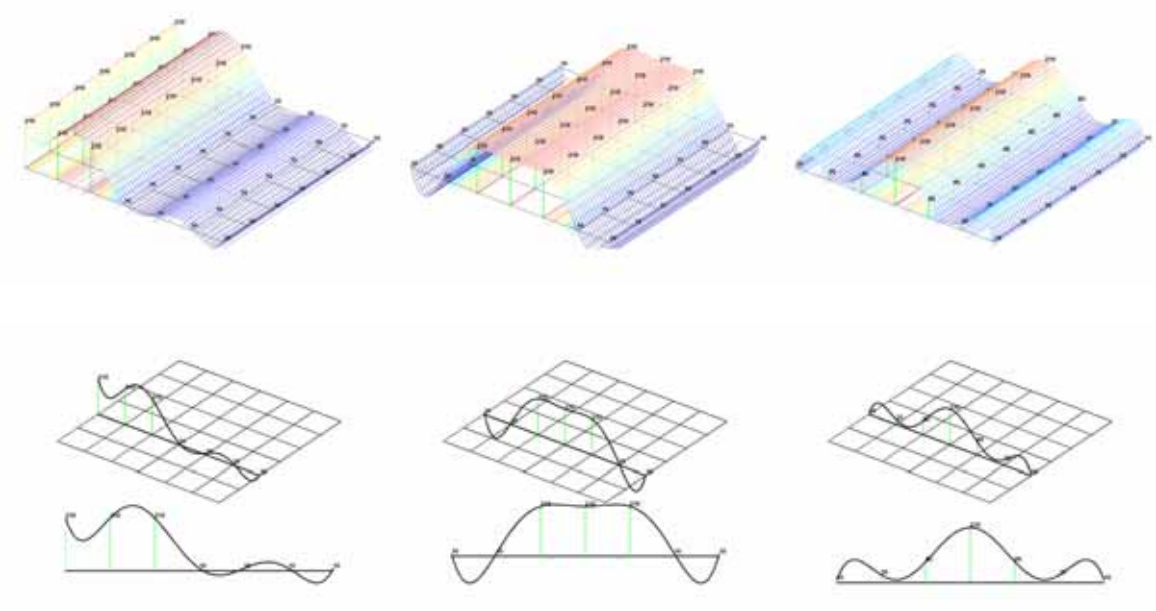
Obrázek 16: Interpolace MQ bázovou funkcí s parametrem ε = 1.0.



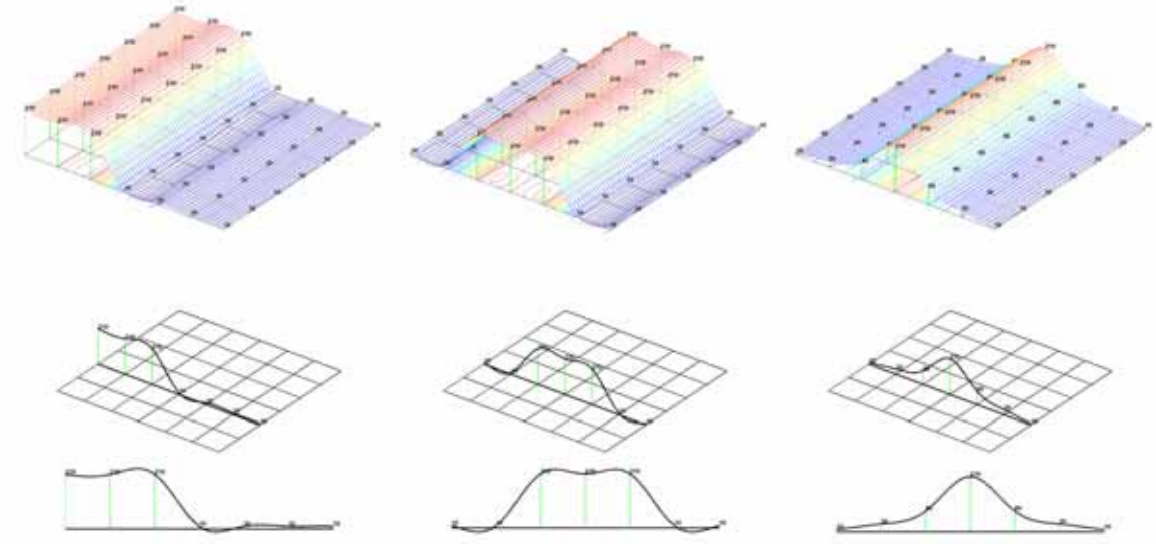


Obrázek 17: Interpolace MQ bázovou funkcí s parametrem ε = 5.0.

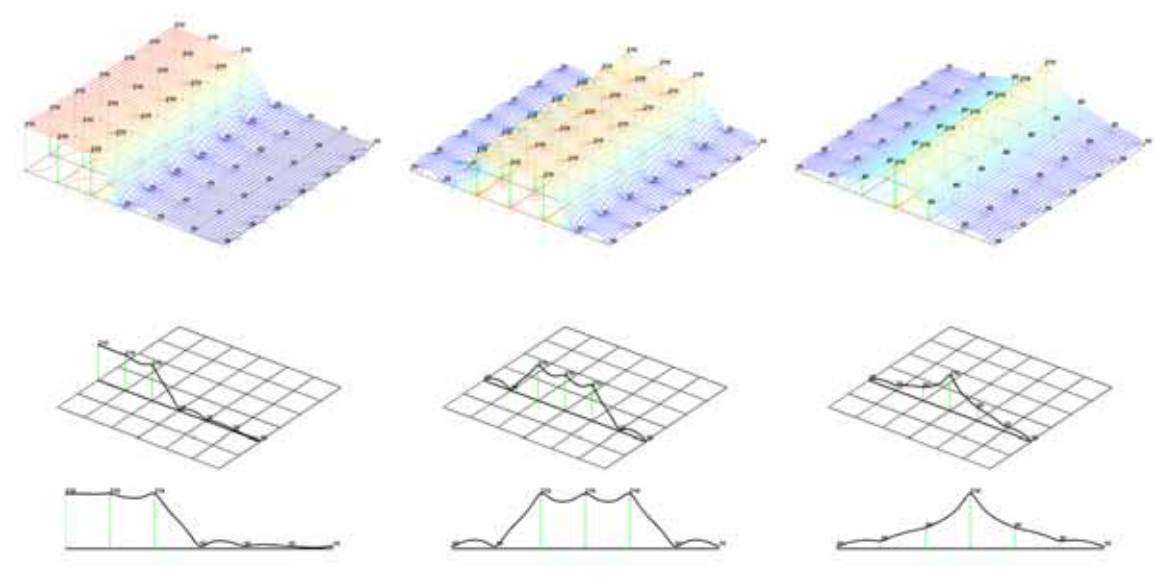
Inverzní multiquadric (IMQ) $\phi(r) = \dfrac{1}{\sqrt{1+(\varepsilon r)^2}}$



Obrázek 18: Interpolace IMQ bázovou funkcí s parametrem ε = 0.1.



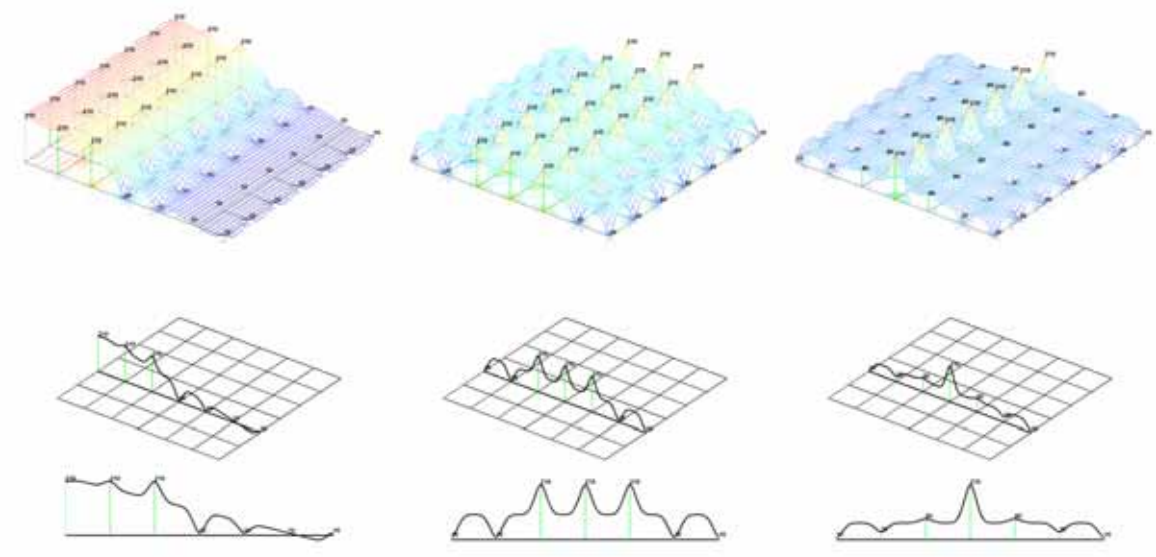
Obrázek 19: Interpolace IMQ bázovou funkcí s parametrem ε = 1.0.



Obrázek 20: Interpolace IMQ bázovou funkcí s parametrem ε = 5.0.

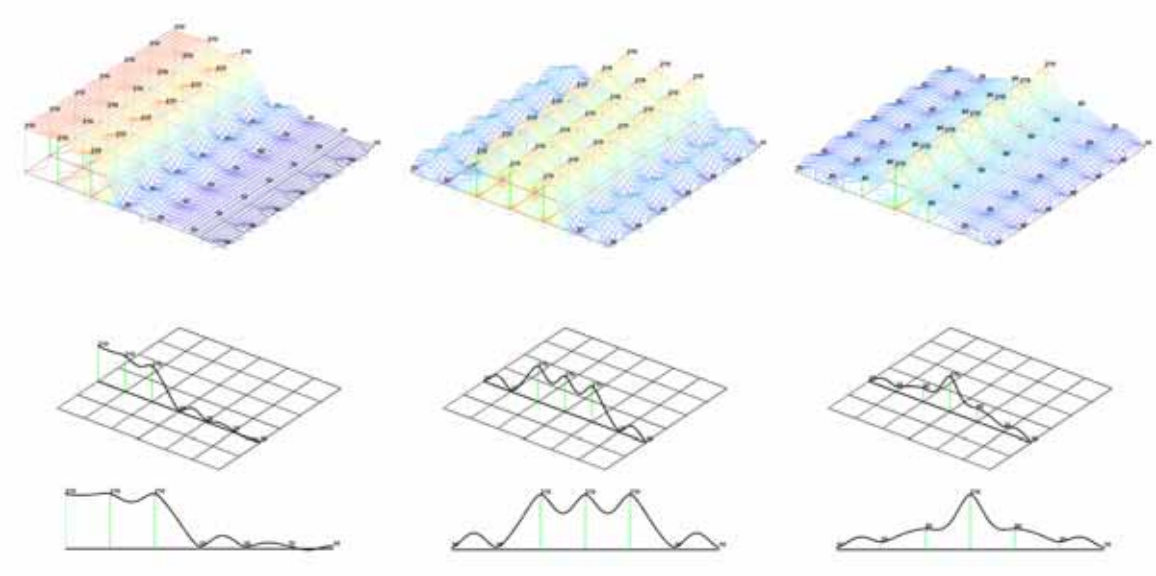
CSRBF $\phi(r) = (1-r)_+^4(4r+1)$

Oblast vlivu CSRBF bázových funkcí je určena parametrem α, jež jsme si popsali v kapitole 2.5 a ukázali na obrázku 2. Následující obrázky ukazují interpolaci při různých volbách volba parametru α.

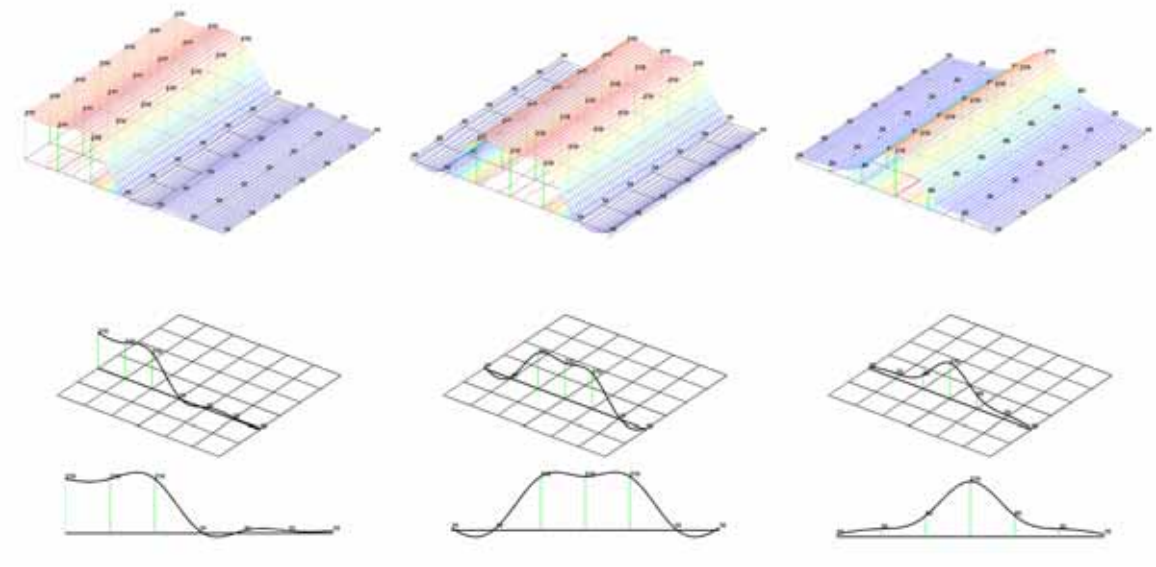


Obrázek 21: Interpolace CSRBF bázovou funkcí s parametrem α = 0.5.

Všimněme si, jak se chová interpolace v případě, jestliže vliv bázové funkce je menší nežli vzdálenosti interpolovaných bodů (vzdálenost je 1). Průběh interpolace prochází body a pak náhle padá ve tvaru bázové funkce k místu, kde prochází rovina definovaní dodatečným polynomem P(x). Tento způsob interpolace je nevhodný a ukazuje k čemu může docházet pokud je parametr α zvolen nevhodně.



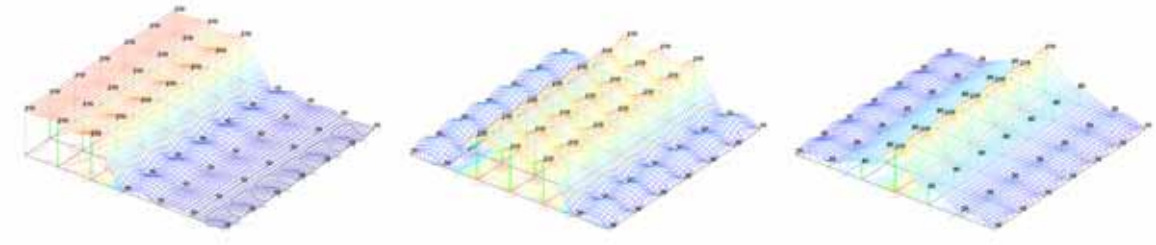
Obrázek 22: Interpolace CSRBF bázovou funkcí s parametrem α = 1.0.

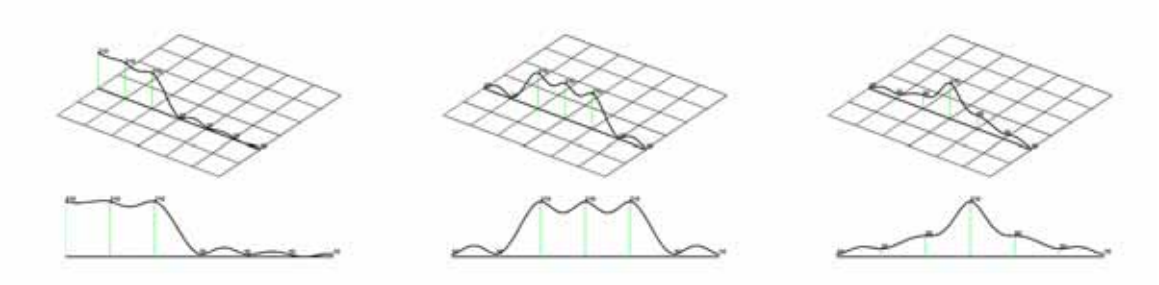


Obrázek 23: Interpolace CSRBF bázovou funkcí s parametrem α = 33.3.

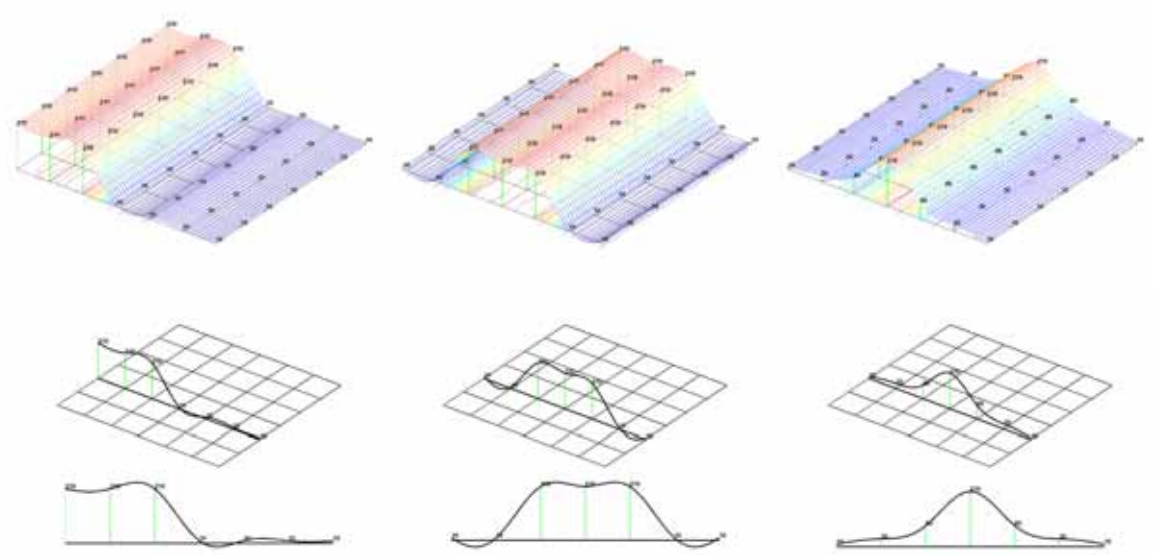
Pokud parametrem α zvětšíme vliv bázové funkce, tak je průběh již mnohem lepší. Musíme si však uvědomit, že parametr α by měl zvětšovat vliv bázové funkce pouze tak, aby sahala přes poškození (díru v obraz), které chceme rekonstruovat. Větším rozsahem nezkazíme nic na rekonstrukci, ale dostaneme plnější matici B, což ovlivní výpočetní náročnost. Degraduje se tedy zásadní výhoda CSRBF bázové funkce.

CSRBF $\phi(r) = (1-r)_+^3(3r+1)$





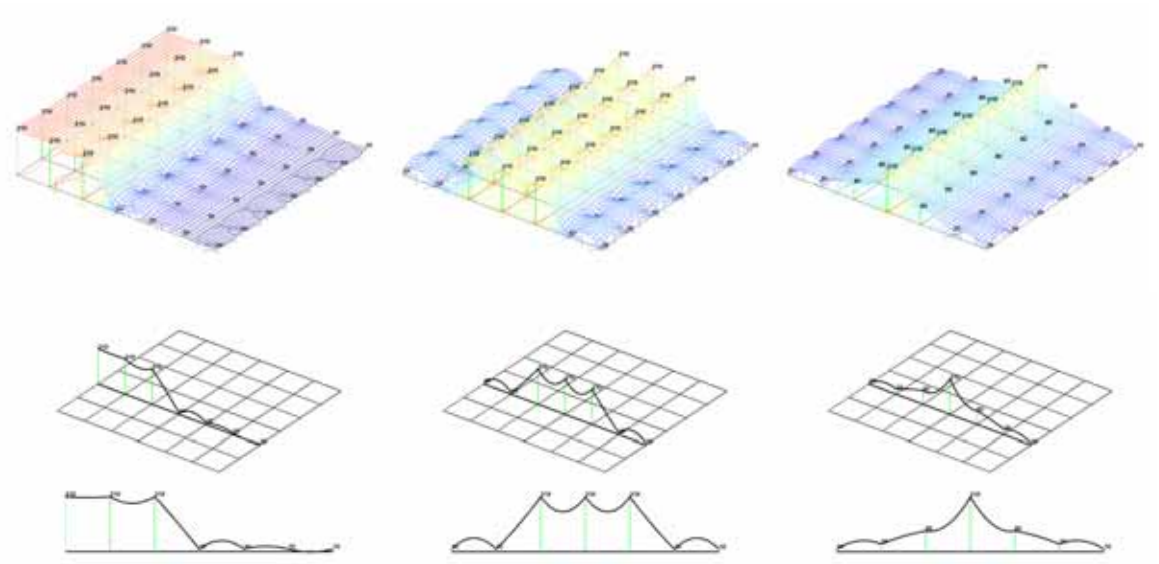
Obrázek 24: Interpolace CSRBF bázovou funkcí s parametrem α = 1.0.



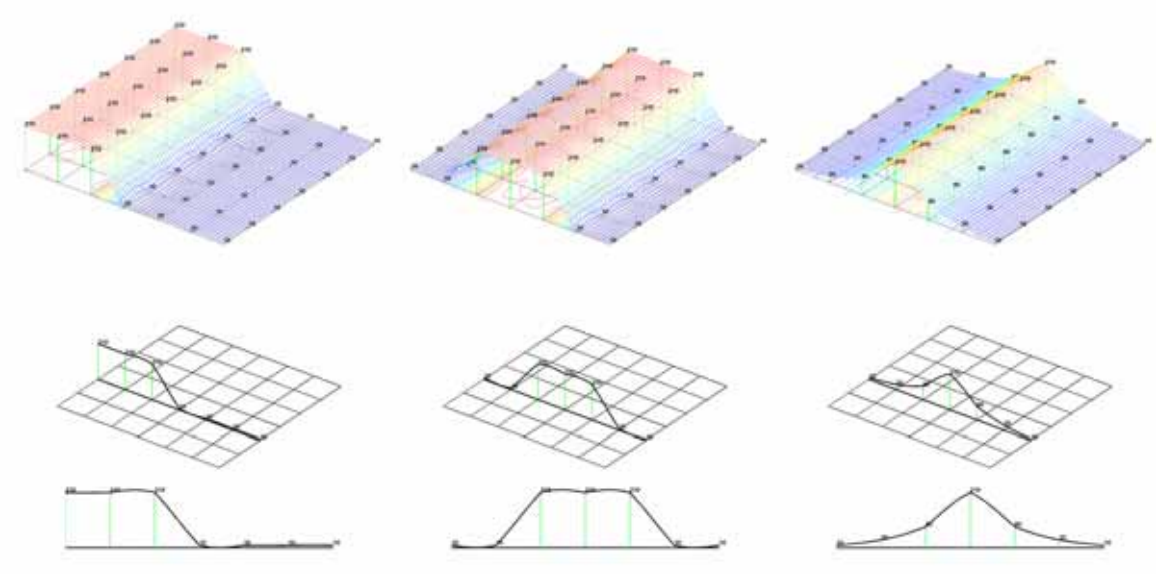
Obrázek 25: Interpolace CSRBF bázovou funkcí s parametrem α = 33.3.

CSRBF $\phi(r) = (1-r)_+^2$

Tato bázová funkce je C^0 spojitá a interpolace touto bázovou funkcí není příliš vhodná.



Obrázek 26: Interpolace CSRBF bázovou funkcí s parametrem α = 1.0.

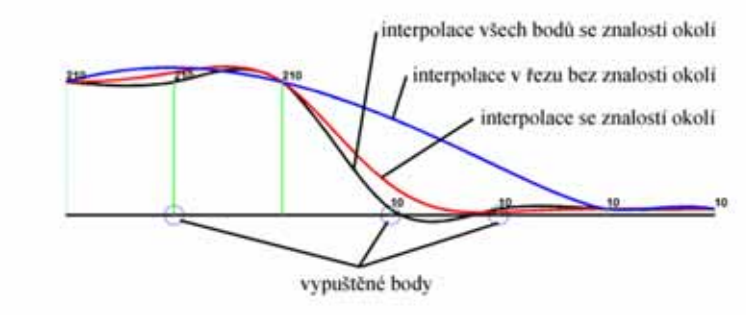


Obrázek 27: Interpolace CSRBF bázovou funkcí s parametrem $\alpha = 33.3$.

PŘÍLOHA C – Rekonstrukce poškození

Příklad:

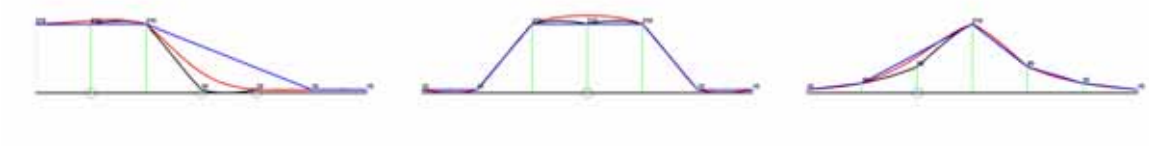
Mějme n-okolí poškozeného obrazového bodu o definované velikosti. Dopočtěme hodnoty poškozených obrazových bodů v oblasti ω_i s použitím různých bázových funkcí. Budeme používat stejná data jako v Příloze B a budeme zobrazovat celkem tři křivky znázorňující průběh povrchu v řezu (Obrázek 28). Vypuštěné body budeme označovat kroužkem.



Obrázek 28: Popis zobrazovaných hodnot.

Lineární bázová funkce $\phi(r) = r$

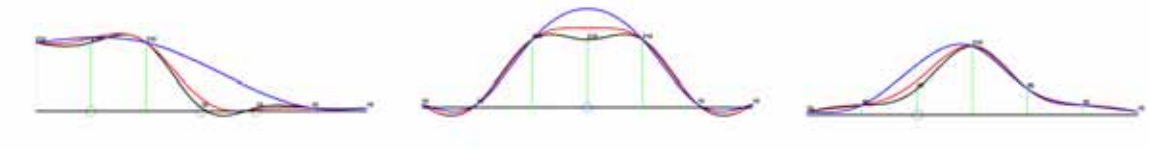
Levý obrázek přesně ukazuje, co se stane pokud nebudeme znát obrazové body na hraně. V případě použití lineární bázové funkce dojde pouze ke spojení dvou krajních obrazových bodů u nichž známe hodnoty (modrá křivka). Pokud k takovému poškození dojde uprostřed obrazových bodů jejichž hodnoty známe, tak tyto obrazové body budou „stahovat“ rekonstruovanou část k původnímu tvaru (červená křivka).



Obrázek 29: Interpolace lineární bázovou funkcí.

Kubická bázová funkce $\phi(r) = r^3$

Zde je pěkně vidět, jak znalost okolních obrazových bodů velmi ovlivňuje výslednou interpolaci.

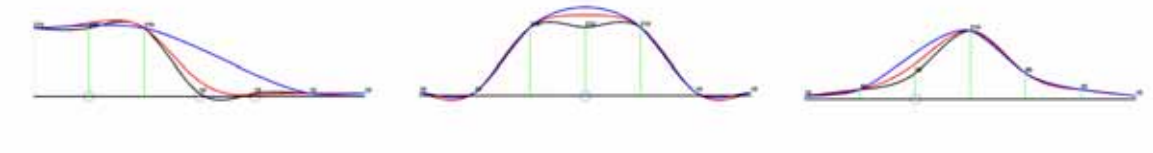


Obrázek 30: Interpolace kubickou bázovou funkcí.

Quantic $\phi(r) = r^5$

Obrázek 31: Interpolace quantic bázovou funkcí.

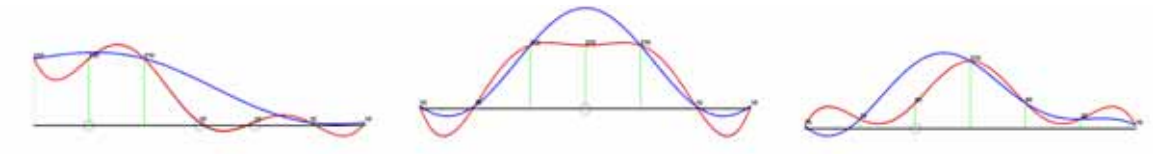
TPS bázová funkce $\phi(r) = r^2 \log r$



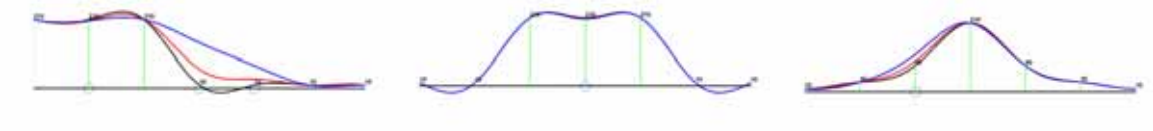
Obrázek 32: Interpolace TPS bázovou funkcí.

GRBF bázová funkce $\phi(r) = e^{-(\varepsilon r)^2}$

Rekonstrukce GRBF bázovou funkcí hodně kmitá, ale určitě je zajímavé, jaká je shoda v případě interpolace všech bodů a interpolace s poškozením. Očividně takto malé poškození nemá na kvalitu interpolace žádný vliv.



Obrázek 33: Interpolace GRBF bázovou funkcí s parametrem ε = 0.1.



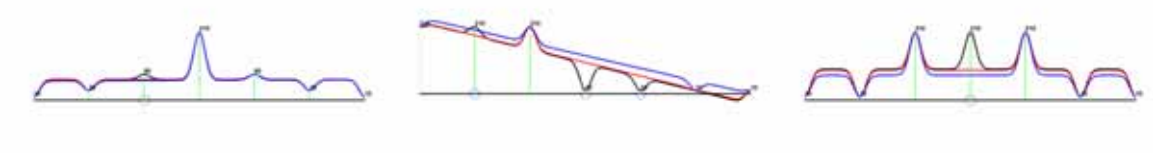
Obrázek 34: Interpolace GRBF bázovou funkcí s parametrem ε = 1.0.

Na Obrázek 34 uprostřed je vidět, jak se prakticky shodují všechny tři interpolace.

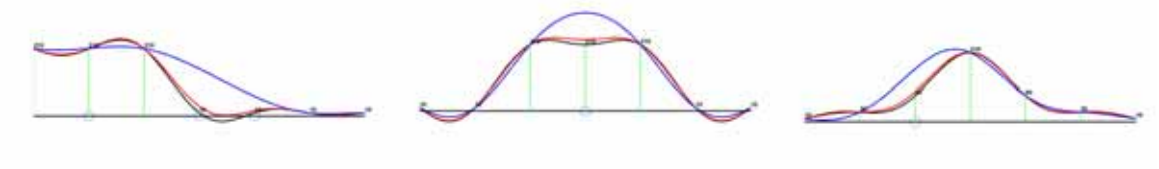
Obrázek 35: Interpolace GRBF bázovou funkcí s parametrem ε = 5.0.

GRBF bázová funkce $\phi(r) = e^{\frac{-r^2}{2\varepsilon^2}}$ [Schagen79]

S parametrem ε = 0.1 je již samotná interpolace nevhodná.

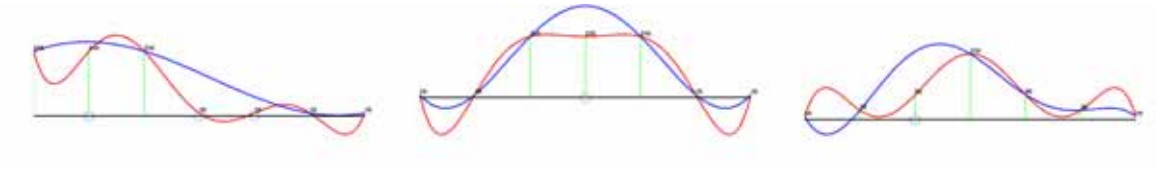


Obrázek 36: Interpolace Schagenovou GRBF bázovou funkcí s parametrem ε = 0.1.



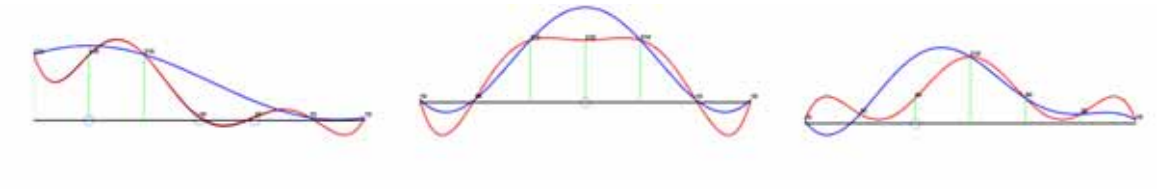
Obrázek 37: Interpolace Schagenovou GRBF bázovou funkcí s parametrem ε = 1.0.

Z uvedených parametrů je ε = 1.0 nejlepší. Interpolace příliš nekmitá a interpolace nepoškozených a poškozených dat jsou si velice blízké.



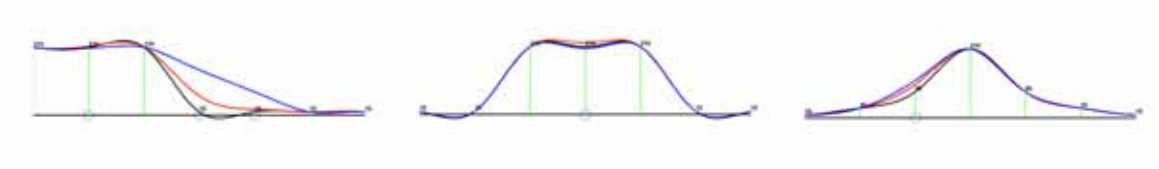
Obrázek 38: Interpolace Schagenovou GRBF bázovou funkcí s parametrem ε = 5.0.

Inverzní kvadratická (IQ) $\phi(r) = \frac{1}{1+(\varepsilon r)^2}$

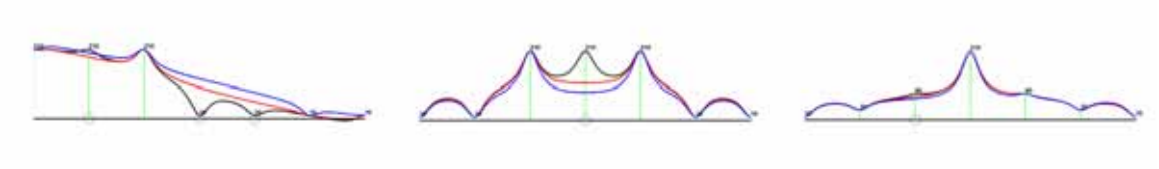


Obrázek 39: Interpolace IQ bázovou funkcí s parametrem ε = 0.1.

V případě, že je parametr menší něž 1, tak interpolace velmi kmitá. Naopak, pokud je parametr hodně velký (např. ε = 5.0), tak je již interpolace nevhodná. Z uvedených parametrů byl nejvhodnější parametr ε = 1.0.



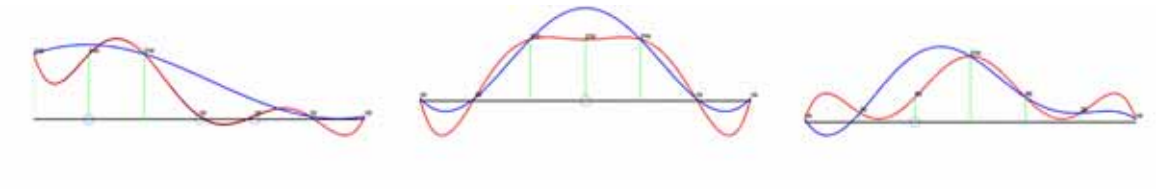
Obrázek 40: Interpolace IQ bázovou funkcí s parametrem ε = 1.0.



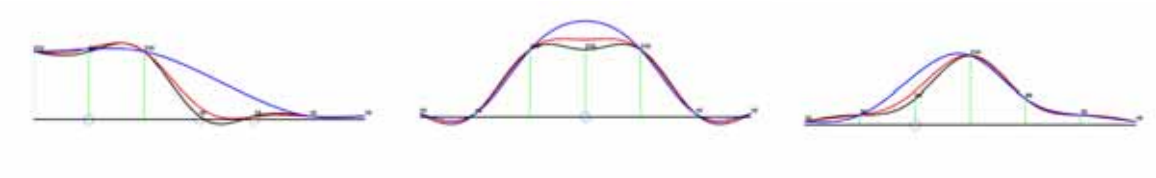
Obrázek 41: Interpolace IQ bázovou funkcí s parametrem ε = 5.0.

Multiquadric (MQ) $\phi(r) = \sqrt{1+(\varepsilon r)^2}$

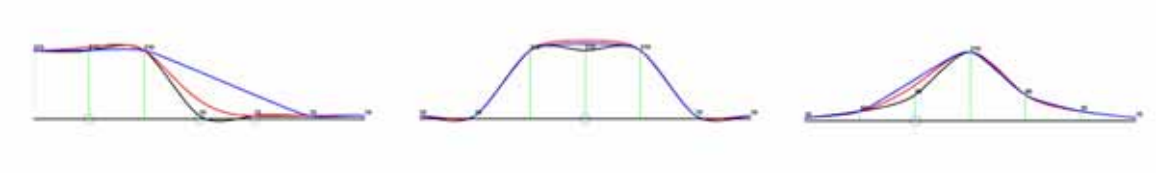
Podobně jako IQ bázová funkce se chová i MQ bázová funkce, avšak v případě velkého tvarovacího parametru má interpolace menší zákmity. Interpolace s parametrem ε = 1.0 se nejvíce přibližuje originálu.



Obrázek 42: Interpolace MQ bázovou funkcí s parametrem ε = 0.1.

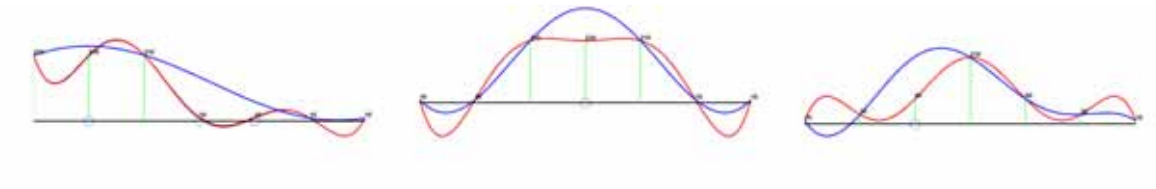


Obrázek 43: Interpolace MQ bázovou funkcí s parametrem ε = 1.0.

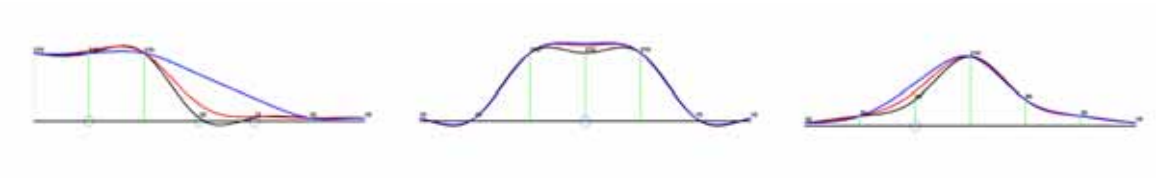


Obrázek 44: Interpolace MQ bázovou funkcí s parametrem ε = 5.0.

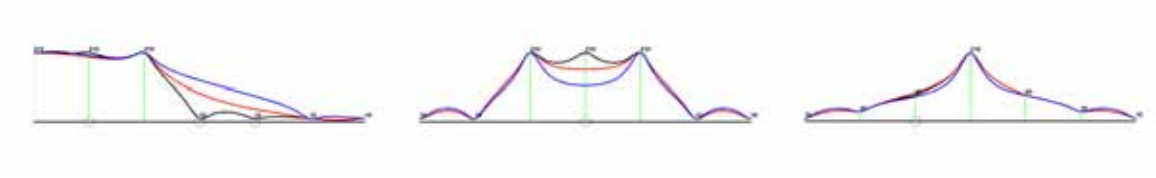
Inverzní multiquadric (IMQ) $\phi(r) = \dfrac{1}{\sqrt{1+(\varepsilon r)^2}}$



Obrázek 45: Interpolace IMQ bázovou funkcí s parametrem ε = 0.1.



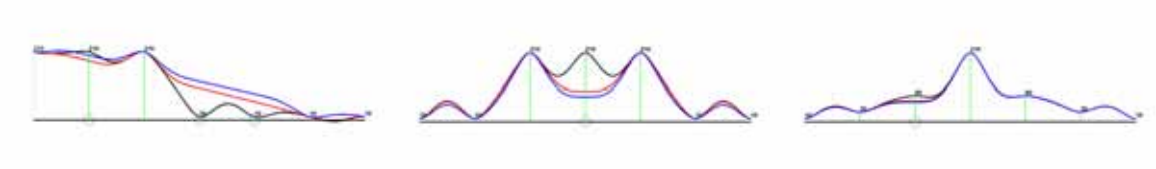
Obrázek 46: Interpolace IMQ bázovou funkcí s parametrem ε = 1.0.



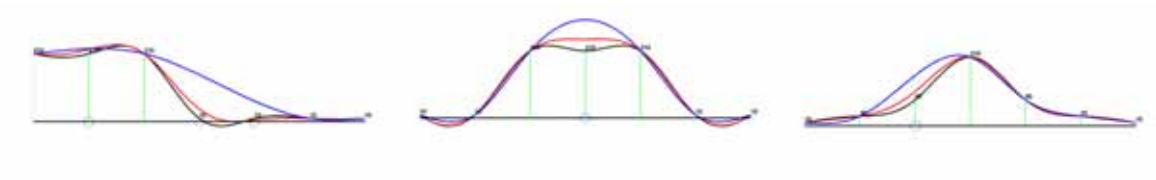
Obrázek 47: Interpolace IMQ bázovou funkcí s parametrem ε = 5.0.

CSRBF $\phi(r) = (1-r)_+^4(4r+1)$

Jak bylo řečeno, tak s malým parametrem α je interpolace nevhodná (Obrázek 48). Pokud parametr upraví bázovou funkci, aby překlenula chybějící body, tak je interpolace v pořádku (Obrázek 49).



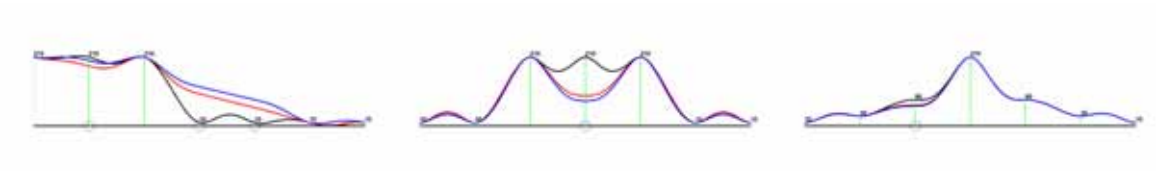
Obrázek 48: Interpolace CSRBF bázovou funkcí s parametrem α = 1.0.



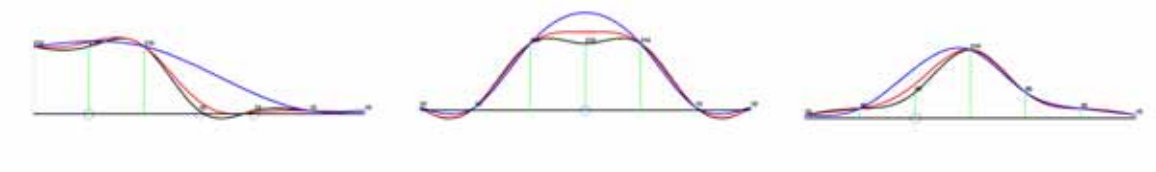
Obrázek 49: Interpolace CSRBF bázovou funkcí s parametrem α = 33.3.

CSRBF $\phi(r) = (1-r)_+^3 (3r+1)$

Bázová funkce je C^2 spojitá a určená pro dimenzi $d = 1$.



Obrázek 50: Interpolace CSRBF bázovou funkcí s parametrem α = 1.0.



Obrázek 51: Interpolace CSRBF bázovou funkcí s parametrem α = 33.3.

CSRBF $\phi(r) = (1-r)_+^2$

Tato bázová funkce se chová obdobně jako lineární bázová funkce.

Obrázek 52: Interpolace CSRBF bázovou funkcí s parametrem α = 1.0.

Obrázek 53: Interpolace CSRBF bázovou funkcí s parametrem α = 33.3.

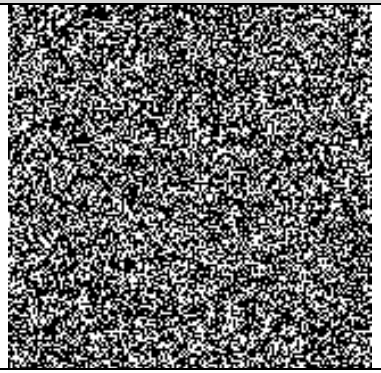
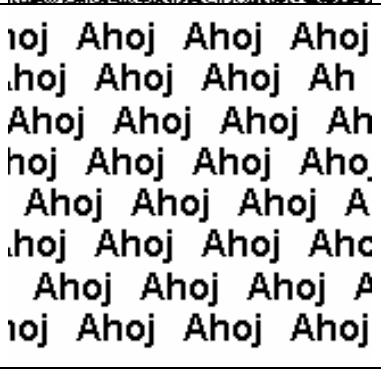
PŘÍLOHA D – Použité bázové funkce

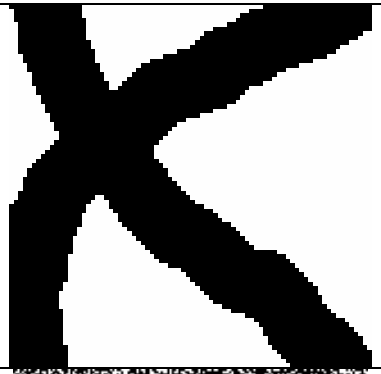
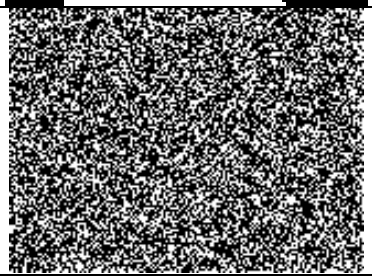
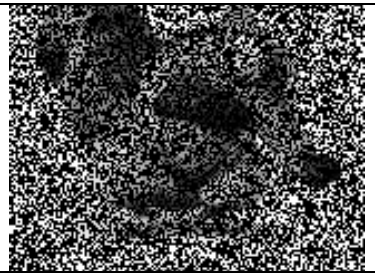
Bázové funkce použité v testech.

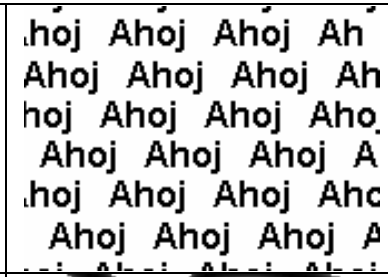
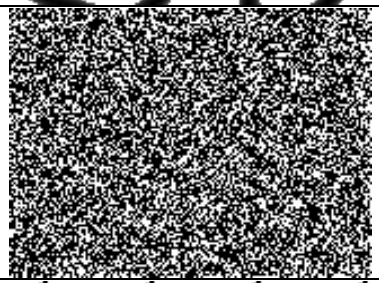
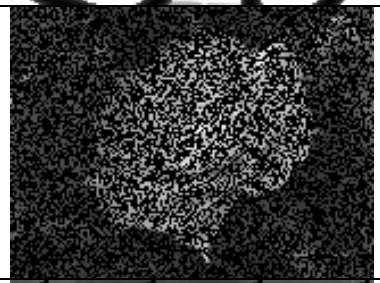
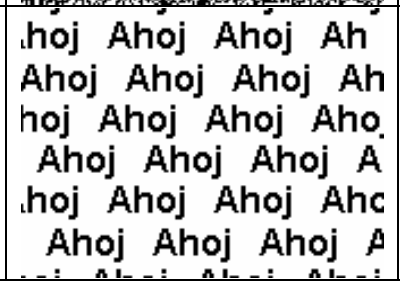
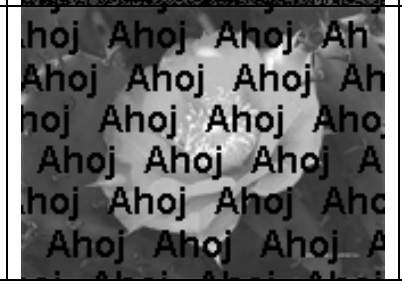
Číslo	Název	Funkce
1	TPS	$\phi(r) = r^2 \log r$
2	Kubická	$\phi(r) = r^3$
3	GRBF	$\phi(r) = e^{-(\varepsilon r)^2}$
4	CSRBF	$\phi(r) = (1-r)_+^4(4r+1)$
5	Kvadratická	$\phi(r) = r^2$
6	Lineární	$\phi(r) = r$
7	CSRBF	$\phi(r) = (1-r)_+^3(3r+1)$
8	IQ	$\phi(r) = \frac{1}{1+(\varepsilon r)^2}$
9	MQ	$\phi(r) = \sqrt{1+(\varepsilon r)^2}$
10	IMQ	$\phi(r) = \frac{1}{\sqrt{1+(\varepsilon r)^2}}$
11	GRBF	$\phi(r) = e^{-\frac{r^2}{2\varepsilon^2}}$
12	Quantic	$\phi(r) = r^5$
13	CSRBF	$\phi(r) = (1-r)_+^2$

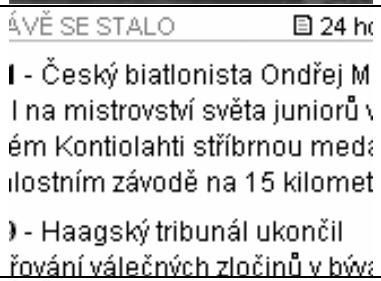
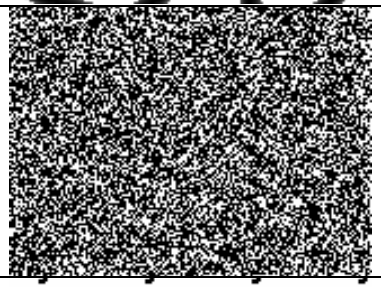
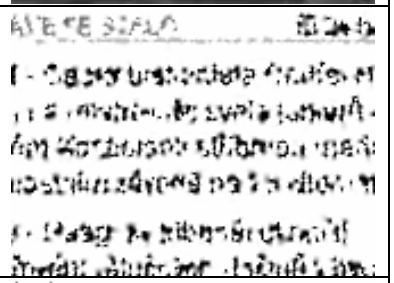
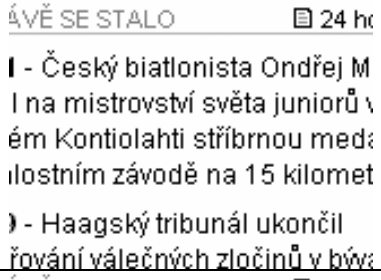
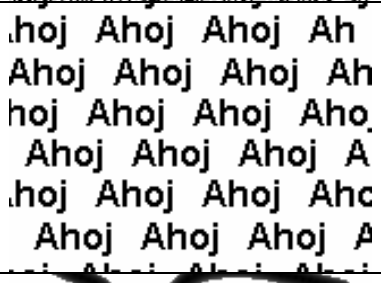
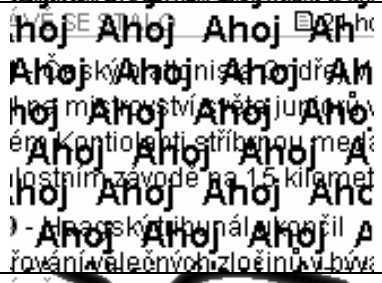
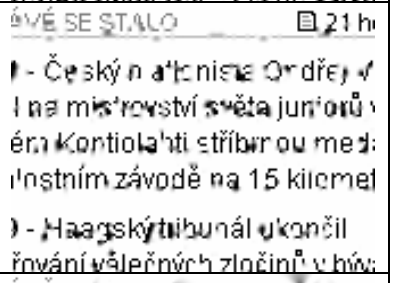
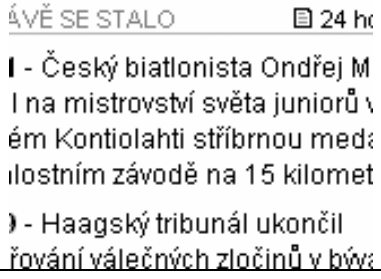
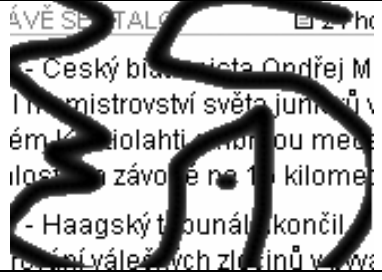
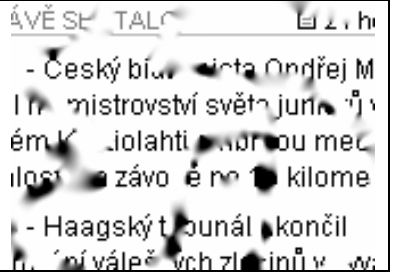
PŘÍLOHA E – Testované obrazy

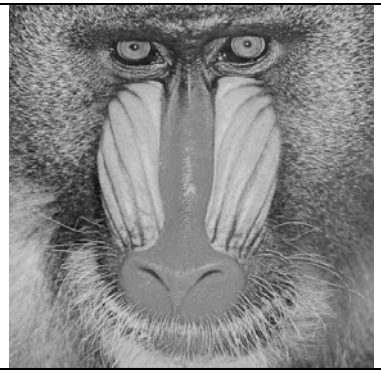
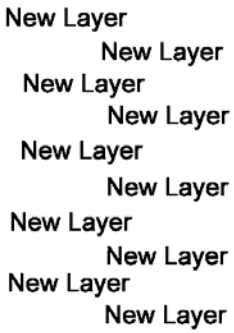
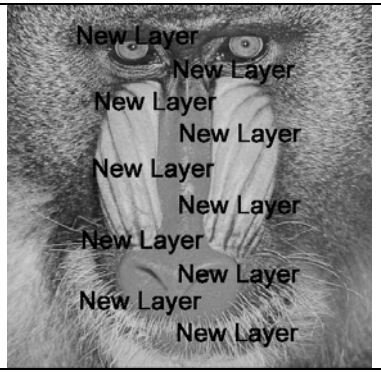
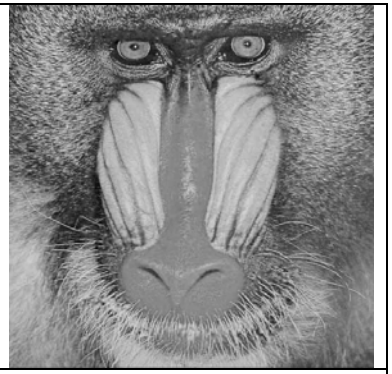
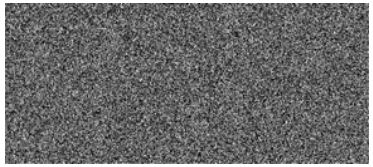
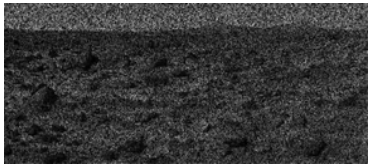
Zde uvádíme seznam testovaných obrazů. Poslední sloupec zobrazuje ukázku rekonstruovaného obrazu pro nejlepší metodu viz. kapitola 3.9.1 Výsledky rekonstrukce jednotlivých metod.

Obraz	Originální obraz	Poškození	Poškozený obraz	Ukázka rekonstrukce
Obraz 1. Lena rozlišení: 160x160 poškození: šum % poškození: 60				
Obraz 2. Lena rozlišení: 160x160 poškození: text % poškození: 19				

Obraz 3. Lena rozlišení: 160x160 poškození: škrábance % poškození: 25				
Obraz 4. Lena (část) rozlišení: 80x80 poškození: škrábance % poškození: 47				
Obraz 5. Scrat rozlišení: 160x120 poškození: šum % poškození: 60				

Obraz 6. Scrat rozlišení: 160x120 poškození: text % poškození: 20				
Obraz 7. Scrat rozlišení: 160x120 poškození: škrábance % poškození: 27				
Obraz 8. Flower rozlišení: 160x120 poškození: šum % poškození: 60				
Obraz 9. Flower rozlišení: 160x120 poškození: text % poškození: 20				

Obraz 10. Flower rozlišení: 160x120 poškození: škrábance % poškození: 27				
Obraz 11. Text rozlišení: 160x120 poškození: šum % poškození: 60				
Obraz 12. Text rozlišení: 160x120 poškození: text % poškození: 20				
Obraz 13. Text rozlišení: 160x120 poškození: škrábance % poškození: 27				

Obraz 14. Baboon rozlišení: 512x512 poškození: text % poškození: 27				
Obraz 15. Mars rozlišení: 905x392 poškození: šum % poškození: 60				

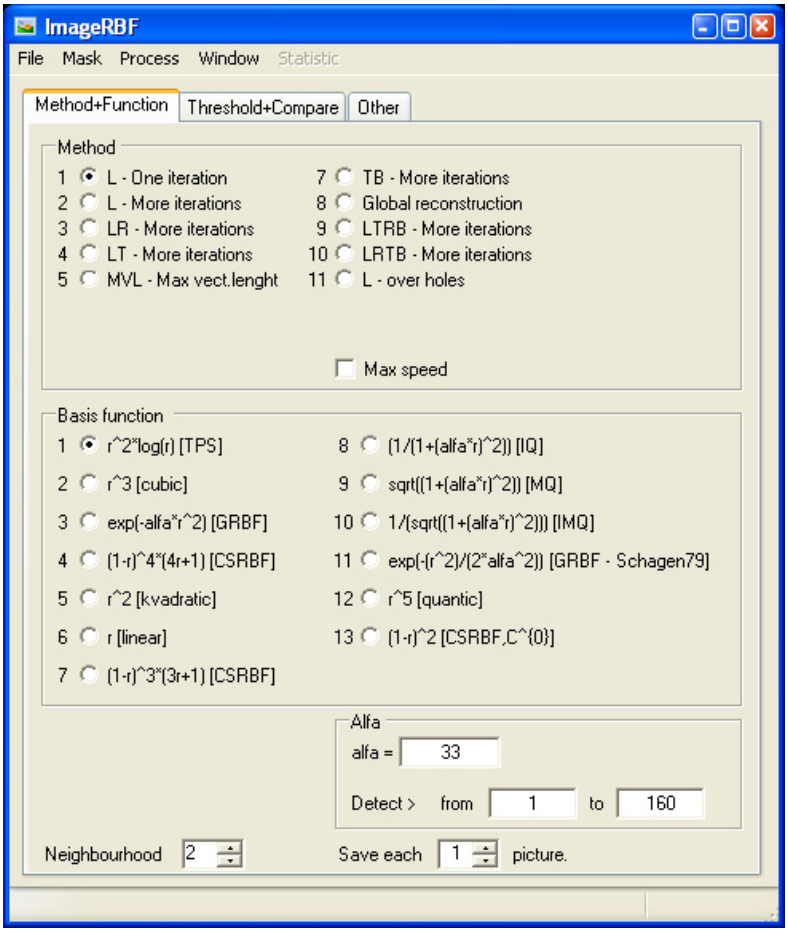
PŘÍLOHA F – Metody

Značeni metod v testech.

Číslo	Název metody	Označení v softwaru	Kapitola
1	Scan-line jednoprůchodová	L – One iteration	3.7.1
2	Scan-line rekonstrukce zleva	L – More iterations	3.7.2
3	Scan-line zleva a zprava	LR – More iterations	3.7.3
4	Scan-line zleva a shora	LT – More iterations	3.7.3
5	Maximální délka vektoru	MLV – Max. vect. lenght	3.7.5
7	Scan-line shora a zdola	TB – More iterations	3.7.3
9	Scan-line zleva, shora, zprava, zdola	LTRM – More iterations	3.7.3
10	Scan-line zleva, zprava, shora, zdola	LRTB – More iterations	3.7.3
11	Scan-line rekonstrukce oboustranná	L – over holes	3.7.4
8	Globální	Global reconstruction	3.6.1

PŘÍLOHA G – Uživatelská dokumentace

Program pro testování různých metod rekonstrukce je tvořen z několika základních oken a záložek, kde můžete měnit nastavení programu. V programu se můžete setkat celkem s třemi různými okny. Jsou to hlavní okno programu (Obrázek), okno zobrazující rekonstruovaný obraz a okno zobrazující průběh rekonstrukce včetně statistiky.



Obrázek:Hlavní okno programu

Okno zobrazující rekonstruovaný obraz je otevřeno ihned po načtení obrazu (pokud není prováděna rekonstrukce podle skriptu) a okno se statistikou je zobrazeno po spuštění rekonstrukce.

Pozor! Rekonstrukce jsou poměrně výpočetně náročné a vzhledem k získání co nejpřesnější informace co se času výpočtu týče, tak bylo upuštěno od implementace metod ve vláknech a může se stát, že se obsah oken nebude věrohodně aktualizovat.

Hlavní menu

Volba	Funkce
File	*hlavní položka menu*
Load	načtení grayscale obrazu, RGB obrazu nebo skriptu
Save	uložení aktuálně zobrazeného obrazu
Exit	ukončení programu
Mask	*hlavní položka menu*
Load	načtení masky

Gener	generování masky
Process	*hlavní položka menu*
Reconstruction	spuštění zvolené rekonstrukce
Comparison	porovnání obrazů za účelem získání hodnot S, S^2, PSNR atd.
Detect best alfa	detekce nejlepší hodnoty alfa
Stop processing	přerušení a ukončení výpočtu
Window	*hlavní položka menu*
Image	zobrazení/zavření okna s rekonstruovaným obrazem
Statistic	zobrazení/zavření okna s průběžnými výsledky rekonstrukce

Záložky
Method+Function

Volba	Funkce
Method	výběr metody rekonstrukce
Max speed	výpočet bude prováděn bez ukládání mezivýsledku a v okně s průběhem rekonstrukce nebudou zobrazovány mezivýsledky
Basis function	volba bázové funkce
Alfa	volba parametru alfa
Detect	volba intervalu pro detekci alfa
Save each	uložení každého n-tého průběžného obrazu rekonstrukce
Neighbourhood	volba parametru určujícího velikost okna
Reconstruction channels	výběr kanálů, které chceme rekonstruovat

Threshold+Compare

Volba	Funkce
Threshold mask	volba prahu pro generování masky pro barevné složky obrazu
Pomparison padding	volba ořízntutí okrajů obrazu, hodnoty S, S^2, PSNR atd. budou vypočteny z ořízntutého obrazu
Threshold for mask	obrazový bod s hodnotou jasu menší než je uvedená hodnota je považován za obrazový bod masky

Other

Volba	Funkce
Out image format	formát ukládaných obrázků
Histogram	zobrazení a zpracování histogramů
Other	obecná nastavení programu

Adresářová struktura:
input – zdrojové obrazy určené k rekonstrukci
mask – masky definující poškození, jež bude rekonstruováno
result – výsleky rekonstrukce (výsledné obrazy)
skript – skripty popisující rekonstrukci
stat – výsledné statistiky

Vstupní soubory:

Základním vstupem programu jsou barevné nebo černobílé obrazy ve formátu JPG, BMP nebo TIFF (adresář *input*). Ke každému obrazu je pak přiřazena maska, která může být v obdobných formátech jako vstupní obraz (adresář *mask*).

Výstupní soubory:
Při rekonstrukci každého obrazu je vytvořen adresář v adresáři *result*. Název adresáře je vytvořen ve formátu:

Datum_Čas_BázováFunkce_VelikostOkna_Alfa_Metoda
(příklad: 2007.11.18_23.29.59_BF1_w2_a0_M1)

V adresáři je vytvořeno několik dalších podadresářů, které obsahují výslednou rekonstrukci. Jsou to adresáře *R*, *G*, *B*, *I*, *RGB* a *diff_im*. V adresářích jsou pak uloženy rekonstruované obrazy včetně poškozeného obrazu a výsledku rekonstrukce. Pokud jsou ukládány i obrazy mezivýsledků, tak jsou taktéž zde. Adresář *R* tedy bude obsahovat minimálně tyto soubory:

*NázevOriginálníhoObrazu*_R_holes.*ext,*
*NázevOriginálníhoObrazu*_R_orig.*ext,*
*NázevOriginálníhoObrazu*_R_result.*ext.*

Výsledná rekonstrukce je v adresáři RGB. Adresář *diff_im* obsahuje chybové obrazy, jak je popsáno v kapitole 3.5. V případě rekonstrukce černobílého obrazu jsou vytvořeny pouze podadresáře *I* a *diff_im*. Výsledky rekonstrukce jsou uloženy v hlavním adresáři. Formát výsledných obrazů je volen v programu v záložce *Other*.

Součástí rekonstrukce jsou i textové soubory s kompletní statistikou. Pro každou rekonstrukci je vytvořen jeden soubor v adresáři *stat* ve formátu:

*Datum_Čas_NázevOriginálníhoObrazu*_stat.txt.

Pokud je provedena rekonstrukce podle skriptu, tak je vytvořen ještě jeden soubor .csv s kompletní statistikou pro všechny rekonstrukce uvedené ve skriptu. Formát názvu souboru je:

*Datum_Čas_NázevSouboruSkriptu*_stat.csv

Vnitřní formát .csv souboru dodržuje standard. Jako oddělovač je definován znak středníku (;) a soubor může být přímo importován do programu MS Excel.
Pro jednu rekonstrukci jsou datum a čas tvořící název souboru shodné.

PŘÍLOHA H – Publikace, pobyty a konference

Publikace

[i] Uhlir, K., Skala, V.: Reconstruction of Damaged Images Using Radial Basis Function, Accepted for publication, EUSIPCO‘2005.

[ii] Uhlir, K., Skala, V.: Radial Basis Function use for the Restoration of Damaged Images, Preseedings of ICCVG‘2004, COMPUTATIONAL IMAGING AND VISION, Kluwer, 2005.

[iii] Uhlir, K., Patera, J., Skala, V.: Radial Basis Function method for iso-line extraction. Electronic computers and informatics, pp. 439-444, VIENALA Press, Košice, September, 2004.

[iv] Uhlir, K., Skala, V.: A survey of method for implicitization of polygonal model , Geometria i Grafika Inzynierska, vol. 6, p.61-70, PL ISSN 1427-9274, 2004.

[v] Uhlir, K.: Modeling methods with implicitly defined objects , University of West Bohemia, Czech Republic, Technical Report No. DCSE/TR-2003-04, 2003.

[vi] Uhlir, K., Skala, V.: The Implicit Function Modelling System - Comparison of C++ and C# Solutions , C# and .NET Technologies'2003, University of West Bohemia, Czech Republic, ISBN 80-903100-3-6, 2003.

[vii] Uhlíř, K., Skala, V.: Kompilovaný HyperFun , Technical Report DCSE/TR-2002-07, University of West Bohemia, Czech Republic, 2002. (Czech language)

[viii] Uhlíř, K., Skala, V. (supervisor): Interaktivní system pro generovaní implicitních funkcí a jejich modelování , Thesis, University of West Bohemia, Czech Republic, 2001.

Pobyty a konference

Pobyty:

12.2.2000 – 15.9.2000 University of Ioannina, Greece, Erasmus/Socrates project

Konference:

04.9.2005 EUSIPCO 2005, Antalya, Turkey. [i]

22.9.2004 ICCVG’2004, Warsaw, Poland. [ii]

12.6.2004 Geometry and Graphics in Teaching Contemporary Engineer , Szczyrk, Poland. [iv]

05.2.2003 C# and .NET Technologies'2003, Czech Republic. [vi]