

# Uporaba radialnih bazičnih funkcij za napovedovanje vrednosti pikselov pri brezizgubnem stiskanju rastrskih slik RGB

Bogdan Lipuš<sup>1</sup>, Sašo Pečnik<sup>1</sup>, Andrej Nerat<sup>1</sup>, Michal Šmolík<sup>2</sup>, Damjan Strnad<sup>1</sup>

<sup>1</sup>Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko

<sup>2</sup>University of West Bohemia, Faculty of Applied Sciences

E-pošta: bogdan.lipus@um.si

## Utilizing Radial Basis Functions for Pixel Value Prediction in Lossless RGB Raster Image Compression

*In this paper, we conduct a study on the application of radial basis functions for predicting pixel values in lossless RGB image compression. The accuracy of these predictions is crucial for achieving optimal image compression ratios. Radial basis functions are utilized within linear system equations to estimate pixel values. We explore different linear equation solvers and examine various types of radial basis functions to analyze their effects on accurate pixel predictions. Our results indicate that the choice of a linear equation solver primarily impacts the reconstruction time, while both the linear equation solvers and radial basis functions have a negligible effect on the accuracy of the reconstruction.*

### 1 Uvod

V članku raziskujemo rekonstrukcijo oziroma napovedovanje vrednosti piksla iz prepoznanih značilnik (skupkov pikselov) v njegovi okolici. Slednje nam lahko služi kot izboljšava pri stiskanju rastrskih slik. Namreč, rezultat uspešne rekonstrukcije je večje število majhnih razlik med napovedano in dejansko vrednostjo pikselov. Pogostejše razlike lahko zapišemo z manjšim številom bitov, kar s pridom izkoriščata znano aritmetično [7] in Huffmanovo kodiranje [8]. V naših eksperimentih smo za zapis razlik med rekonstruirano in dejansko vrednostjo uporabili aritmetično kodiranje. Kot značilke za rekonstrukcijo smo izbrali piksele v okolici piksla, ki ga rekonstruiramo, oziroma izračunavamo razliko med napovedano in dejansko vrednostjo. V osnovi se osredotočamo na brezizgubno stiskanje (kodiranje dejanskih razlik), ga pa želimo tudi razširiti na skoraj brezizgubno ter izgubno stiskanje (kjer pa ni nujno, da kodiramo dejanske razlike, ampak dovoljujemo približke v smislu boljšega stiskanja podatkov).

### 2 Postopek napovedovanja pikselov

Za napovedovanje vrednosti piksla smo uporabili radialne bazične funkcije [1], ki se uporabljajo pri interpolaciji in aproksimaciji podatkov [2, 3, 4, 5]. Interpolacijska shema je določena z naslednjo interpolacijsko funkcijo:

$$f(\mathbf{x}) = \sum_{i=1}^n \lambda_i \phi(|\mathbf{x} - \mathbf{c}_i|) + ax + by + c, \quad (1)$$

$$\sum_{i=1}^n \lambda_i c_i^x = \sum_{i=1}^n \lambda_i c_i^y = \sum_{i=1}^n \lambda_i = 0. \quad (2)$$

kjer predstavljajo vrednosti  $[\lambda_1, \dots, \lambda_n, a, b, c]$  rešitev sistema linearnih enačb.  $\phi$  je radialna bazična funkcija,  $n$  je število pikselov v bližnji okolici piksla  $\mathbf{x}$ , ki mu določamo vrednost,  $[\mathbf{c}_i]$  so koordinate teh pikselov. Iz enačbe 1 lahko zapišemo naslednjo enačbo:

$$f(\mathbf{c}_i) = h_i \quad (3)$$

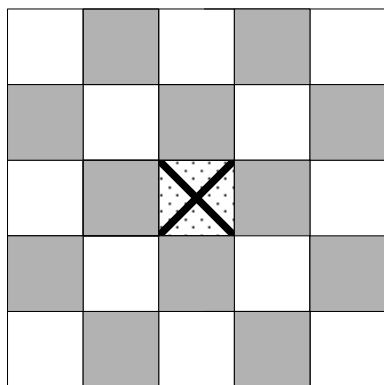
Skupaj z enačbo 2 zgradimo linearni sistem enačb (enačba 4), kjer so  $\phi_{i,j} = \phi(|\mathbf{c}_i - \mathbf{c}_j|)$ ,  $i, j = 1, \dots, n$  ter  $h_i$  so vrednosti okoliških pikselov na koordinatah  $\mathbf{c}_i$ . Rešitev sistema linearnih enačb je nabor parametrov  $(\lambda_1, \lambda_2, \dots, \lambda_n, a, b, c)$  za vsako komponento RGB posebej. Posamezen nabor uporabimo za izračun vrednosti posamezne komponente RGB piksla na vhodnih koordinatah  $\mathbf{x}$ .

Bližnja okolica piksla je opredeljena s procesirnim oknom, ki je kvadrat velikosti  $N$ . Na sliki 1 je primer kvadrata velikosti 5. V središču je piksel (označen z  $\mathbf{X}$ ) za katerega napovedujemo njegovo vrednost iz (znanih) pikselov v njegovi okolici (označenih s sivo barvo).

#### 2.1 Procesiranje rastrske slike

Napovedovanje vrednosti pikselov izvedemo tako, da se s procesirnim oknom (slika 1) pomikamo iz levega zgornjega roba do desnega spodnjega roba obravnavane slike velikosti  $w \times h$ . Znotraj procesirnega okna imamo piksele, ki bodo uporabljeni pri rekonstrukciji (označeni s sivo barvo). Z belo so označeni pikseli, ki smo jih bodisi rekonstruirali v prejšnjih položajih procesirnega okna, bodisi bodo še rekonstruirani z nadaljnjim pomikom procesirnega okna. V primeru na sliki 1 rekonstruiramo središčni piksel. Ko je središče procesirnega okna na vogalih ali robovih rastrske slike se del procesirnega okna nahaja zunaj območja rastrske slike. V tem primeru kot rekonstruirano (napovedano) vrednost vzamemo kar povprečje vrednosti (znanih) pikselov (označenih s sivo barvo) okolici središčnega piksla znotraj območja rastrske slike.

$$\begin{bmatrix} \phi_{1,1} & \phi_{1,2} & \dots & \phi_{1,n} & c_1^x & c_1^y & 1 \\ \phi_{2,1} & \phi_{2,2} & \dots & \phi_{2,n} & c_2^x & c_2^y & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \phi_{n,1} & \phi_{n,2} & \dots & \phi_{n,n} & c_n^x & c_n^y & 1 \\ c_1^x & c_2^x & \dots & c_n^x & 0 & 0 & 0 \\ c_1^y & c_2^y & \dots & c_n^y & 0 & 0 & 0 \\ 1 & 1 & \dots & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \dots \\ \lambda_n \\ a \\ b \\ c \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ \dots \\ h_n \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad (4)$$



Slika 1: Procesirno okno (5 × 5)

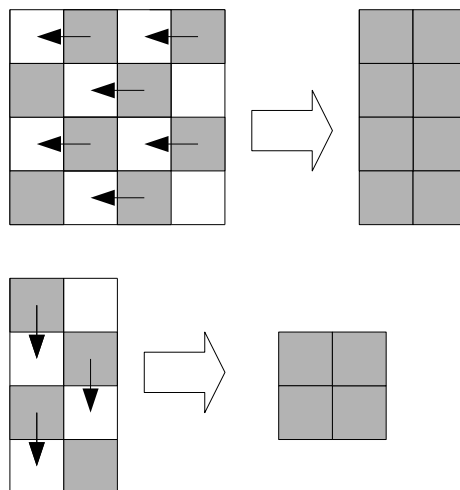
Tabela 1: Radialne bazične funkcije.

Funkcija	Enačba
Linearna	$1/x$
Kubična	$2x^3 - 3x^2 + 1$
Wyvill	$-\frac{4}{9}x^6 + \frac{17}{9}x^4 + \frac{22}{9}x^2 + 1$
Wyvill-2	$-x^6 + 3x^4 + 3x^2 + 1$
Kosinus	$\frac{1}{2} \cos(\pi x) + 1$
Kvadratna	$(1 - x)^2$

Po izračunu vseh napovedi oziroma razlik med rekonstruirano in dejansko vrednostjo zložimo piksele v novo rastrsko sliko, katere velikost je  $w/2 \times h$  (glej postopek na sliki 2). Torej imamo  $1/2(wh)$  rekonstruiranih pikselov in enako število preostalih. Postopek ponovimo na tako dobljeni rastrski sliki, in dobimo še končnih  $1/4(wh)$  pikselov ter rastrsko sliko, katere velikost  $w/2 \times h/2$  je za polovico manjša (slika 2). Postopek bi se seveda lahko rekurzivno ponavljali na na tako dobljeni manjši rastrski sliki. Skupaj imamo torej  $3/4(wh)$  rekonstruiranih pikselov in četrtno preostalih pikselov.

## 2.2 Radialne bazične funkcije

Tabela 1 prikazuje radialne bazične funkcije, ki smo jih uporabili pri eksperimentih. Funkcije so določene na intervalu med  $x = 0$  in  $x = 1$ . Vhodni interval radialne bazične funkcije prilagodimo velikosti  $N$  procesirnega okna.



Slika 2: Postopek rekonstrukcije

## 2.3 Reševanje sistema linearnih enačb

V eksperimentih smo uporabili metode reševanja sistema linearnih enačb, prikazane v tabeli 2. Razlikujejo se glede na natančnost in hitrost [6]. V nadaljevanju bomo raziskali, kako se te lastnosti izražajo pri rekonstrukciji rastrskih slik oziroma, ali imajo vpliv na čas in rekonstrukcijo pikselov na rastrski sliki.

Tabela 2: Lastnosti različnih metod za reševanje sistema linearnih enačb, povzetih iz [6]

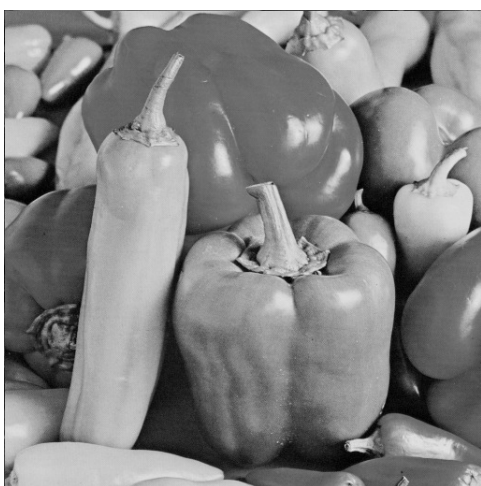
Metoda	Hitrost	Natančnost
PartialPivLU	++	+
FullPivLU	-	+++
HouseholderQR	++	+
ColPivHouseholderQR	+	+++
FullPivHouseholderQR	-	+
CompleteOrthogonal-Decomposition	+	++
LLT	+++	+++
LLTD	+++	+++

## 3 Rezultati

V eksperimentih smo uporabili testni rastrski sliki 3 in 4. Obe sta velikosti  $512 \times 512$  s tremi barvnimi komponentami RGB. Torej imamo skupaj  $3/4(wh) = 196.608$  rekonstruiranih pikselov oziroma 589.824 razlik (upoštevajoč vse tri komponente RGB).



Slika 3: Lenna



Slika 4: Pepper

Najprej smo izmerili čase rekonstrukcije glede na izbrano metodo reševanja sistema linearnih enačb [6]. Tabela 3 kaže skupni čas rekonstrukcije 196.608 pikselov v skladu s postopkom, predstavljenim v poglavju 2. Eksperimente in izračune smo izvajali na računalniku Intel® Core™ i9-12900KF 64 GB RAM.

Kot najhitrejša metoda za reševanje sistema linearnih enačb se izkaže metoda LLT, najslabša pa je metoda CompleteOrthogonal-Decomposition, ki je skoraj trikrat počasnejša. Tabela 4 prikazuje razporeditev razlik med napovedano vrednostjo ter dejansko vrednostjo. V izračun so zajete vse tri komponente RGB. Rezultati kažejo, na minimalne razlike pri napovedovanju vrednosti. Natančno ujemanje med rekonstruirano in dejansko vrednostjo je 8,75% primerih (od 51.607 do 51.626), odstopa zgolj najhitrejša metoda reševanja sistema linearnih enačb LLT, ki je nekoliko slabša od preostalih metod (8,65%). Skoraj dve tretjini absolutnih razlik je manjših od 5, preostala tretjina je večja ali enaka 5. Frekvenca pojavljanja večjih razlik se velikostjo absolutnih razlik zmanjšuje. Slednje pomeni, da lahko manjše razlike zapišemo s krajšimi kodami, za večje razlike pa uporabimo daljše kode.

Na osnovi frekvence pojavljanja posameznega sim-

Tabela 3: Časi rekonstrukcije pikselov v ms glede na izbrano metodo reševanja sistema linearnih enačb (uporabljena je linearna radialna bazična funkcija)

Metoda	Lenna (v ms)	Pepper (v ms)
PartialPivLU	308	312
FullPivLU	458	510
HouseholderQR	599	591
ColPivHouseholderQR	664	659
FullPivHouseholderQR	321	313
CompleteOrthogonal-Decomposition	681	677
<b>LLT</b>	<b>244</b>	<b>242</b>
LLTD	301	295

bolov (to je razlik med napovedano in dejansko vrednostjo) lahko izračunamo povprečno število bitov za kodiranje vseh razlik po naslednji enačbi (pri čemer bi v prvotni obliki vsak simbol zapisali z 8 biti):

$$b_p = \sum -\log_2 p_i, \quad (5)$$

kjer so  $p_i = [0, 1]$  frekvence pojavljanja simbolov (v našem primeru razlik med rekonstruirano in dejansko vrednostjo).

Tabela 4: Razporeditev izračunanih absolutnih razlik  $r$  med rekonstruirano oziroma napovedano in dejansko vrednostjo posameznih pikselov na rastrski sliki Lenna ter  $b_p$  - povprečno število bitov za zapis razlike  $r$  (uporabljena je linearna radialna bazična funkcija)

Metoda	Lenna		
	$r = 0$	$r < 5$	$b_p$
PartialPivLU	51.626	322.853	4,13
FullPivLU	51.614	322.849	4,13
HouseholderQR	51.613	322.853	4,13
ColPivHouseholderQR	51.619	322.853	4,13
FullPivHouseholderQR	51.626	322.853	4,13
CompleteOrthogonal-Decomposition	51.619	322.853	4,13
LLT	51.022	<b>322.390</b>	4,19
LLTD	<b>51.607</b>	322.847	4,13

Razporeditev izračunanih absolutnih razlik  $r$  med rekonstruirano oziroma napovedano in dejansko vrednostjo posameznih pikselov na testnih rastrskih slikah prikazujeta tabeli 5 in 6. Kot najboljša se izkaže uporaba kvadratne radialne bazične funkcije, s katero dosežemo 10,01% (ali 59.020 razlik je enako vrednosti 0) (Lenna) oziroma 8,75% (ali 51616 razlik je enako 0) ujemanj med rekonstruirano in dejansko vrednostjo pikselov. Najslabša se pri obeh metodah izkaže radialna bazična funkcija Wyvill-2. Je pa razvidno, tako kot tudi pri vseh ostalih metodah stiskanja, da so nekatere rastrske slike predstavljajo večji izziv za stiskanje podatkov.

Tabela 5: Razporeditev izračunanih absolutnih razlik  $r$  med rekonstruirano in dejansko vrednostjo posameznih pikslov na rastrski sliki Lenna ter  $b_p$  - povprečno število bitov za zapis razlike  $r$  (uporabljena metoda reševanja sistema linearnih enačb je bila PPL - PartialPivLU)

Funkcija	Lenna		
	$r = 0$	$r < 5$	$b_p$
Linearna	59.028	389.878	4,13
Kubična	58.225	386.688	4,14
Wyvill	57.503	384.793	4,15
Wyvill2	56.780	381.605	4,18
Kosinus	58.187	386.922	4,14
Kvadratna	<b>59.020</b>	389.877	<b>4,13</b>

Tabela 6: Razporeditev izračunanih absolutnih razlik  $r$  med rekonstruirano in dejansko vrednostjo posameznih pikslov na rastrski sliki Pepper ter  $b_p$  - povprečno število bitov za zapis razlike  $r$  (uporabljena metoda reševanja sistema linearnih enačb je bila PPL - PartialPivLU)

Funkcija	Pepper		
	$r = 0$	$r < 5$	$b_p$
Linearna	51.626	322.853	4,42
Kubična	50.435	318.079	4,49
Wyvill	50.290	316.740	4,51
Wyvill2	49.677	314.050	4,52
Kosinus	50.503	318.483	4,49
Kvadratna	<b>516.16</b>	322.846	<b>4,42</b>

## 4 Zaključek

V članku smo raziskali uporabo radialnih bazičnih funkcij v sistemih linearnih enačb. Uporabili smo različne metode za reševanje sistema linearnih enačb in radialne bazične funkcije. Rezultati so pokazali, da imajo različne metode reševanja sistema linearnih enačb predvsem vpliv na čas rekonstrukcije rastrske slike, nimajo pa posebnega vpliva na natančnost rekonstrukcije. Tudi izbira radialne bazične funkcije ima neznamenit vpliv na čas rekonstrukcije, ker so vrednosti radialnih bazičnih funkcij izračunane vnaprej glede na znano velikost okna, in kasneje zgolj uporabljene pri vsaki rekonstrukciji posameznega piksla v okviru procesirnega okna.

V nadaljevanju raziskav se bomo osredotočili predvsem na izdelavo reverzibilne sheme stiskanja, ter optimalne izbire pikslov v okolici piksla, ki ga rekonstruiramo oziroma iščemo razlike med napovedano in dejansko vrednostjo piksla. V teku raziskav se je tudi pokazalo, da poseben izziv predstavljajo izraziti prehodi med različnimi vrednostmi barv na rastrski sliki ali pa, da v okolici ni primernih pikslov za rekonstrukcijo piksla. Skušali bomo tudi raziskati, kako se s primerno izbiro pikslov izogniti napakam, ki se pojavljajo v teh območjih rastrskih slik.

## 5 Zahvala

Projekt (Paradigma stiskanja podatkov z odstranjevanjem obnovljivih informacij, št. J2-4458) sta financirali Javna agencija za znanstvenoraziskovalno in inovacijsko dejavnost Republike Slovenije ter Czech Science Foundation (Project No. 23-04622L) iz državnega proračuna.

## Literatura

- [1] M. D. Buhmann (2003). *Radial Basis Functions*, Cambridge University Press.
- [2] N. Kozhokin, V. V. Savchenko, M. Senin, I. Hagiwara (2003). An approach to surface retouching and mesh smoothing, *The Visual Computer*, 19(7-8), 549–564.
- [3] W. Wang, X. Qin (2006). An image inpainting algorithm based on CSRBF interpolation, *International Journal of Information Technology*, 12 (6), 112–119.
- [4] K. Uhlř, V. Skala (2006). Radial basis function use for the restoration of damaged images, V: K. Wojciechowski, B. Smolka, H. Palus, R. Kozera, W. Skarbek, L. Noakes (uredniki), *Computer Vision and Graphics International Conference*, Springer, Vol. 32, 839–844.
- [5] B. Lipuš, B. Žalik (2012). Efficient reconstruction of images with deliberately corrupted pixels, *Informatika*, 23(1), 47–63.
- [6] G. Guennebaud, B. Jacob in ostali (2023). Eigen v3, <https://eigen.tuxfamily.org>.
- [7] I. H. Witten, M. N. Radford, in J. G. Cleary (1987). Arithmetic coding for data compression. *Communications of the ACM*, 30(6) : 520-540.
- [8] D. A. Huffman, (1952). A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9), 1098-1101.