Zbornik radova Fakulteta tehničkih nauka, Novi Sad

**UDK: 621.39**
**DOI: https://doi.org/10.24867/17BE06Markovic**

# GENERATOR KODA ZA OPTIMALNU IMPLEMENTACIJU BINDCT TRANSFORMACIJE

# CODE GENERATOR FOR OPTIMAL IMPLEMENTATION OF BINDCT TRANSFORM

Miloš Marković, *Fakultet tehničkih nauka, Novi Sad*

**Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

**Kratak sadržaj –** *U ovom radu biće predstavljeni DCT, BinDCT* i *HEVC. Prikazaćemo postupak transformacije leptira u transformaciju od tri koraka podizanja za N=4. Izložićemo glavni problem i dati rešenje za novi kod generator, njegove prednosti u odnosu na prethodno rešenje i rezultate njegovog korišćenja.*

**Ključne reči:** *DCT, BinDCT kod generator, HEVC, Binarna DCT, three lifting steps transformacije*

**Abstract** – *In this paper will be presented DCT, BinDCT, and HEVC. We will show the procedure of butterfly transformation into a transformation of three lifting steps for N = 4. We will present the main problem and give a solution for the new code generator, its advantages in relation to the previous solution, and the results of its use.*

**Keywords:** *DCT, BinDCT code generator, HEVC, Binary DCT, three lifting steps transformations*

## 1. UVOD

Kompresija je često neophodna kako bi se omogućilo i olakšalo uvođenje novih tehnologija radi smanjenja potrebnog vremena za skladištenje i obradu podataka. U [1] predstavljen je novi kodek za kompresiju slika sa 32 b/p zasnovan na HEVC intra-kodovanju. Rad koristi BinDCT transformaciju [2] sa fleksibilnim veličinama blokova, koje variraju od 4×4 do 32×32 piksela; 12 metoda intra-predikcije, JPEG-XR univerzalnu šemu kvantizacije (sa povećanim rasponom parametara kvantizacije), i malo modifikovan CABAC na kraju, za efikasno entropijsko kodovanje.

Rezultati iz rada [3] značajno nadmašuju rezultate ranije predstavljenih šema kompresije. Analizom radova [2,3,4] došli smo na ideju da napravimo kod generator za binDCT transformaciju. U radu [4] opisana je binDCT transformacija sa *fiksnim parametrima*. Motivacija za ovaj rad proističe iz činjenice da u postojećem kodu za BinDCT iz rada [4] nije bilo moguće da parametri binDCT transformacije budu *promenljivi* kako bi korisnik mogao da utiče na njenu implementaciju.

Problem je rešen tako što je napravljen univerzalni *kod generator* (dodatni program), koji generiše potrebni kod i omogućava da unošenjem *p, u, p* parametara dobijamo različite konfiguracije za binDCT i izaberemo najbolju.

Na taj način moguće je odrediti optimalnu konfiguraciju, što u ranijim implementacijama binDCT nije bio slučaj. Parametri se unose ručno u shifts.xml fajl, proizvoljnim izborom i na osnovu tih parametara dobijaju se rezultati za tačno potrebnu vrednost parametra određene transformacije, što pre nije bilo omogućeno. Nakon parsiranja .xml fajla i podešavanja stanja, promenljivih i vrednosti promenljivih, kod generator čita template fajl (šablon). Kada pronađe ključne reči za generisanje koda i iskoristi vrednosti p, u i p parametara koji su prethodno isparsirani generiše novi C kod odnosno generiše neophodne .cpp i .h fajlove.

Generisani fajlovi se potom mogu koristiti za tehnike kompresije zasnovane na binDCT, kao što je rad [4].

**NAPOMENA:**
**Ovaj rad proistekao je iz master rada čiji mentori su bili: doc. dr Branko Brkljač i dr Miloš Radosavljević.**

## 2. HIGH EFFICIENCY VIDEO CODING (HEVC)

HEVC, poznat i kao H.265 i MPEG-H 2 Part 2, trenutno je najnoviji i najnapredniji video standard, koji su zajednički razvili Video Coding Experts Group (VCEG) i Moving Picture Experts Group (MPEG), okupljeni oko jedne grupe Joint Collaborative Team on Video Coding (JCT-VC). U poređenju sa AVC-om, HEVC nudi od 25% do 50% bolju kompresiju podataka pri istom nivou video kvaliteta, ili značajno poboljšan kvalitet videa pri istoj brzini protoka. Podržava rezolucije do 8192 × 4320, uključujući 8K UHD. HEVC koristi celobrojne DCT i DST transformacije sa različitim veličinama blokova između 4×4 i 32×32 piksela. Standard HEVC definiše sintaksu ili format komprimovane video sekvence i metod dekodovanja. Stvarni dizajn kodera nije standardizovan. Podržava visoko fleksibilno razdvajanje video sekvence.

## 3. DISCRETE COSINE TRANSFORM

Discrete Cosine Transform ili DCT predstavlja diskretnu transformaciju koja se pokazala kao veoma pogodna za kompaktan zapis i spektralnu analizu signala, tako što očuvava ukupnu energiju signala u malom broju značajnih težinskih koeficijenata. U opštem slučaju definiše se korišćenjem aritmetike sa realnim brojevima. DCT je našla primenu u skoro svim algoritmima kompresije kao što su na primer: JPEG, MPEG, H.26X, MP3.

DCT kompresija je poznata i kao blok kompresija. Koristi blokove raznih veličina piksela počevši od 4×4 pa do 32×32. Mnogi različiti brzi algoritmi za izračunavanje DCT-a, razvijeni su za primenu na fotografije i video zapise. Neki algoritmi baziraju se na retkim faktorizacijama DCT matrice, i mnogi od njih su rekurzivni. Kada je u pitanju obrada slika i video zapisa, da bi se podaci efikasno komprimovali neizostavno je potrebna kvantizacija. Diskretna kosinusna transformacija diskretne funkcije *f(j), j=0,1,⋯,N-1* definisana je kao:

$$F(k) = \frac{2c(k)}{N}\sum_{j=0}^{N-1} f(j)cos\left[\frac{(2j+1)k\pi}{2N}\right]; k = 0,1,\cdots,N-1 \qquad (1)$$

Slično, inverzna diskretna kosinusna transformacija određena je sa:

$$f(j) = \sum_{k=0}^{N-1} c(k)F(k)cos\left[\frac{(2j+1)k\pi}{2N}\right]; j = 0, 1, \cdots, N-1 \qquad (2)$$

$$c(k) = \begin{cases} \frac{1}{\sqrt{2}}, & k = 0 \\ 1, & k = 1, 2, \cdots, N-1 \end{cases}$$

Transformacija poseduje svojstvo visokog energetskog zbijanja (kompakcije). Takođe poseduje i svojstvo linearnosti, što omogućava laku primenu teorije linearnih sistema. Brzi računski algoritam za izračunavanje DCT opisan je u radu [3] gde se nalaze sve formule i računski koraci formiranja matrice [F].

## 4. BINARY DCT (BINDCT)

Diskretna kosinusna trasformacija (DCT) uspešno je primenjena na kodovanje slika visoke rezolucije. Konvencionalna metoda implementacije DCT-a koristila je brzu Furijeovu transformaciju dvostruke veličine (2D FFT), koja koristi složenu aritmetiku tokom izračunavanja. Fast Discrete Cosine Transform – FDCT predstavlja efikasniji algoritam koji uključuje samo realne operacije za brzu diskretnu kosinusnu transformaciju skupa od N tačaka. Algoritam se može proširiti na bilo koju željenu vrednost ako je $N = 2^m, m \geq 2$.

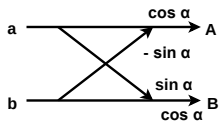
Generalizacija se sastoji u naizmeničnim leptirima matrice kosinusa i sinusa sa binarnim matricama za uređivanje elemenata matrice u oblik koji čuva prepoznatljiv obrnuti redosled bita (engl. bitreverse), raspored (engl. pattern) na svakom drugom čvoru. Generalizacija nije jedinstvena, otkriveno je nekoliko alternativnih metoda ali je metoda koja koristi $\frac{3N}{2}log_2N - 1 + 2$ realnih sabiranja i $Nlog_nN - \frac{3N}{2} + 4$ realnih množenja najjednostavnija za razumevanje. Ovakva metoda je približno šest puta brža od konvencionalnog pristupa koji koristi 2D FFT.

## 5. APROKSIMACIJA BRZOG RAČUNSKOG ALGORITMA

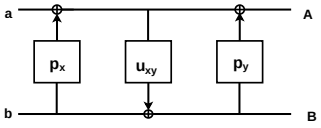
Aproksimacija brzog računskog algoritma pomoću p i u blokova objašnjena je u radu [2]. Na osnovu rada [2] možemo objasniti kako se leptir sa slike 1(a) u radu [2] prevodi u leptir koji je pogodan za transformaciju od tri koraka podizanja odnosno konverziju na p i u blokove. Oblik leptira koji je neophodno da bude ispoštovan kako bi bila dozvoljena transformacija od tri koraka podizanja izveden je iz formule (1) koja se nalazi u radu [2] i prikazan na slici 1. Prikazan leptir sa slike 1, može se konvertovati na sledeći blok dijagram od tri koraka podizanja prikazan na slici 2. Na datom blok dijagramu mogu se primetiti promenljive $p_x, u_{xy}, p_y$ gde $x$ i $y$ predstavljaju indekse *p*, *u* i *p* blokova koji se koriste u transformaciji.

Pored promenljivih mogu se primetiti vrednosti signala *a* i *b* koji predstavljaju ulazne signale u blok za transformaciju, kao i signale *A* i *B* koji predstavljaju vrednosti izlaznih signala iz bloka za transformaciju.

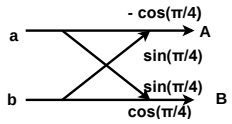
Transformacija od tri koraka podizanja ključna je za binDCT kod generator. Na slici 3 biće prikazan leptir koji se nalazi na slici 1(a) u radu [2], dok se na slici 4 nalazi konvertovan leptir koji je pogodan za transformaciju od tri koraka podizanja.



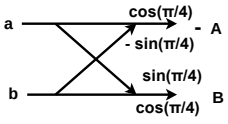

Slika 1 Prikaz osnovnog leptira „three lifting steps“ transformacije




Slika 2 Blok za transformaciju od tri koraka podizanja



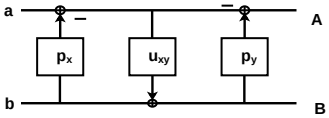

Slika 3 Leptir koje je rezultat postupka DCT transformacije




Slika 4 Konvertovan leptir sa slike 3

Neophodno je da leptir sa slike 3 konvertujemo u leptir oblika kao što je prikazan na slici 1. Konačno, leptir sa slike 4 možemo da prikažemo kao blok dijagram transformacije od tri koraka podizanja. Ovaj blok dijagram može se videti na Fig. 1. (a) iz rada [2].

Elementi $p_4, u_4$ i $p_5$ koji su prikazani na pomenutoj slici iz rada [2] predstavljaju elemente našeg blok dijagrama.

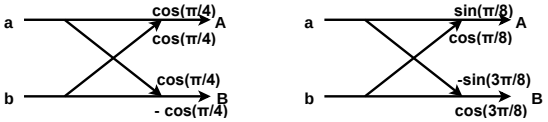



Slika 5 Prikaz transf. od tri koraka podizanja leptira sa slike 4

Prethodno opisan postupak transformacije leptira u transformaciju od tri koraka podizanja ključan je za binDCT kod generator koji je zasnovan na ovoj transformaciji.

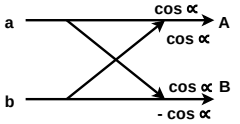
## 6. TRANSFORMACIJA N = 4

Ova transformacija sastoji se od dva leptira kao što je prikazano na slici 2 u radu [3]. Za ovu transformaciju možemo reći da imamo samo jedan prolaz odnosno stanje transformacije. Na narednoj slici biće prikazani leptiri tog stanja.




Slika 6 Prikaz leptira za transformaciju 4 × 4

Prvi leptir sa slike 6 može se prikazati kao na slici 7:




Slika 7. Prikaz prvog leptira transf. 4 × 4 u generalnom obliku

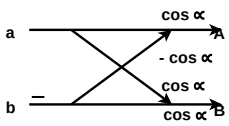
Potrebno je leptir sa slike 7 prevesti u oblik sa slike 1, pogodan za transformaciju od tri koraka podizanja. Leptir sa slike 7 možemo zapisati kao sledeći sistem jednačina

$$\begin{aligned} A &= a\cos\alpha + b\cos\alpha \\ B &= a\cos\alpha + b(-\cos\alpha) \end{aligned} \qquad (3)$$

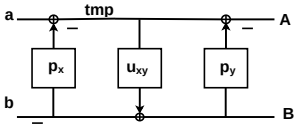
Zbog minusa koji može izaći ispred u drugoj jednačini, menja se izraz u prvoj jednačini na sledeći način:

$$\begin{aligned} A &= a\cos\alpha - b(-\cos\alpha) \\ B &= a\cos\alpha - b\cos\alpha \end{aligned} \qquad (4)$$

Ovi izrazi mogu se predstaviti kao leptir sa slike 8.




Slika 8 Prvi leptir transf. 4x4 u transf. obliku




Slika 9 Prikaz transf. od tri koraka podizanja leptira sa slike 8

Leptir sa slike 8 u potpunosti odgovara leptiru prikazanom na slici 1. Konačno možemo napisati transformaciju od tri koraka podizanja leptira prikazanog na slici 8. Na slici 9 može se primetiti da postoje dva dodatna minusa u dijagramu, usled:

$$p = \frac{\cos\frac{\pi}{4} - 1}{\sin\frac{\pi}{4}} = -0{,}41421;\ u = \sin\frac{\pi}{4} = 0{,}70710 \qquad (5)$$

Na osnovu slike 9 možemo napisati potrebne izraze transformacije, odnosno inverzne transformacije.

$$\begin{aligned} tmp &= a + b \gg p_x \\ B &= tmp \gg u_{xy} - b \\ A &= tmp - B \gg p_y \end{aligned} \qquad (6)$$

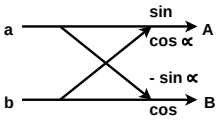
$$\begin{aligned} tmp &= A + B \gg p_y \\ b &= tmp \gg u_{xy} - B \\ a &= tmp - b \gg p_x \end{aligned} \qquad (7)$$

U prethodnim izrazima može se primetiti operator $\gg$ koji nije standardni izraz za „pomeranje bitova udesno“ već predstavlja izraz koji kod generator koristi za označavanje zbira levih šiftova „<<“ pomerenih u desno „>>“.

Na primer, ako je $p_y = \frac{25}{32}$, *calculate_number_of_shifts* u kod generatoru vratiće: (4~3~0) „/“ 5, tj. broj pomeranja koji odgovara izrazu: $(2^4+2^3+2^0)/2^5$. Pošto u izrazu (7) imamo *B* koji se pomera za $p_y$ kod generator će liniju u šablonu zameniti sledećom linijom koda:

*tmp = A + (((B << 4) + (B << 3) + B) >> 5)* (8)

Preostali (drugi) leptir sa slike 6 za koji je potrebno izvršiti transformaciju možemo prikazati kao:




Slika 10 Prikaz drugog leptira transf. 4 × 4 u opštem obliku

Potrebno je leptir sa slike 10 prevesti u oblik sa slike 1 koji je pogodan za transformaciju od tri koraka podizanja. Leptir sa slike 10 možemo zapisati kao:

$$\begin{aligned} A &= a\cos\alpha + b\sin\alpha \\ B &= a(-\sin\alpha) + b\cos\alpha \end{aligned} \qquad (9)$$

Zbog minusa koji je izašao ispred u drugoj jednačini menja se izraz u prvoj jednačini na sledeći način:

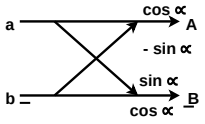
$$\begin{aligned} A &= a\cos\alpha - b(-\sin\alpha) \\ -B &= a\sin\alpha - b\cos\alpha \end{aligned} \qquad (10)$$

Prethodni izraz odgovara leptiru na slici 11, a koji odgovara leptiru prikazanom na slici 1. Konačno, možemo napisati transformaciju od tri koraka podizanja leptira. Na osnovu slike 12 sledi da su:

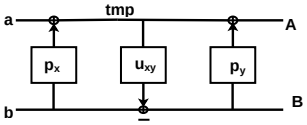
$$\begin{aligned} tmp &= a + b \gg p_x \\ B &= b - tmp \gg u_{xy} \\ A &= tmp + B \gg p_y \end{aligned} \qquad (11)$$

$$\begin{aligned} tmp &= A - B \gg p_y \\ b &= tmp \gg u_{xy} + B \end{aligned} \qquad (12)$$

$$a = tmp - b \gg p_x$$




Slika 11 Drugi leptir transf. 4 × 4




Slika 12 Prikaz transf. od tri koraka podizanja leptira sa slike 11

## 7. OPIS KOD GENERATORA

Zadatak kod generatora jeste da na osnovu vrednosti parametra *p* i *u* elemenata generiše C kod, odnosno da se generišu .cpp i .h fajlovi. Programsko rešenje je pisano u Python programskom jeziku. Konfiguracioni fajl za unos *p* i *u* parametara nalazi se u programskom rešenju na sledećoj lokaciji ../Configuration/shifts.xml.

Konfiguracioni fajl se sastoji od sledećih xml elemenata:

- <shifts> - predstavlja element u kojem se nalaze vrednosti za *p* i *u* elemente određenog stanja za transformacije 4×4
- <shift_N×N class-name=”ShiftN×N”> - predstavlja element određene transformacije gde je N = 4
- <stage value=“x“> - predstavlja stanje pojedine transformacije
- <px> - predstavlja vrednost *p* elementa određene transformacije
- <ux> - predstavlja vrednost *u* elementa određene transformacije.

Glavna prednost shifts.xml fajla jeste u tome što korisnik može ručno da unosi željene vrednosti parametara čime dobija rezultate za tačno njemu potrebnu vrednost parametra određene transformacije što mu pre nije bilo omogućeno. Programsko rešenje počinje svoje izvršavanje u fajlu BinDCT_CodeGenerator.py u kojem se nalazi main metoda. Unutar ovog fajla importuju se neophodni moduli koji su pisani unutar ovog programskog rešenja. Moduli koji su importovani jesu: *MyParser as MyP, MyCodeGenerator as MyCG, MyClass as Class, time. CalculateShifts* modul ima zadatak da izračuna vrednosti levih i desnih šiftova uz pomoć calculate_right_shifts (number) i calculate_left_shifts (number) metode. Glavni zadatak ovog modula jeste da u okviru calculate_number_of_shifts (value) metode na osnovu vrednosti parametra value vrati vrednosti levih/desnih šiftova za taj parametar što možemo videti u izrazu (8).

*CalculateShifts* modul ima zadatak da izračuna vrednosti levih i desnih „šiftova“ uz pomoć calculate_right_shifts (number) i calculate_left_shifts(number) metode.

Glavni zadatak ovog modula jeste da u okviru calculate_number_of_shifts (value) metode na osnovu vrednosti parametra value vrati vrednosti levih/desnih šiftova za taj parametar što možemo videti u izrazu (8). MyCodeGenerator.py modul ima zadatak da pročita šablon koji se nalazi na lokaciji /Template/Template.cpp, čitajući liniju po liniju ovog šablona-a, proveri da li linija sadrži ključnu reč za generisanje novog koda i da na osnovu ključne reči generiše potreban kod. Nakon generisanja koda, novokreirani kod je potrebno da upiše u fajl TComTrBinDCT.cpp na lokaciji ../Generated.

## 8. EKSPERIMENTALNA POSTAVKA

Potrebno je napraviti par različitih konfiguracija za BinDCT transformaciju od 4×4 tačke po koracima podizanja i odrediti *p, u, p* vrednost za svaki korak zadane transformacije. Primer parametara konfiguracije za jedan *p* parametar dat je u okviru tabele 1. Vrednost ugla

transformacije određujemo tako što na osnovu izračunatog $p$ , $u$ , $p$ parametra tražimo najpribližniju vrednost našeg rezultata formule i beležimo koji ugao pripada tom rezultatu. Tako su uglovi za transformaciju od 4×4: $\frac{\pi}{4}$ i $\frac{3\pi}{8}$.

TABELA 1 Primer parametara konfiguracije

| | Floating-point | C1 | C2 | C3 | C4 |
|---|---|---|---|---|---|
| **$P_1$** | 0,41421 | $\frac{13}{32}$ | $\frac{7}{16}$ | $\frac{3}{8}$ | $\frac{5}{16}$ |

Potrebno je dati konačne rezultate i izgled za par nama bitnih konfiguracija. C1 predstavlja najpribližniju konfiguracionu vrednost, dok C4 predstavlja jednu od „najudaljenijih“ konfiguracionih vrednosti. Kod generator na osnovu dobijenih konfiguracija treba da generiše rezultate na osnovu kojih možemo da analiziramo da li postavljeni parametri konfiguracije zadovoljavaju bitne vrednosti performansi i brzine kompresije.

## 9. REZULTATI

Za dobijanje rezultata iskorišćen je kodek koji je prethodno bio dostupan u okviru rada [1] i u kojem se inače koristi BinDCT transformacija. Pomenuti koder koristio je All-Intra (All-I) konfiguraciju. Performanse kodeka su procenjene korišćenjem standardnih RD krivih. Prosečna razlika između dve RD krive data je u obliku Bjøntegaard delta PSNR mere (BD-PSNR). PSNR predstavlja razliku između dve vertikalno poravnate (interpolirane) tačke iz različitih RD krivih. Relativno vreme kodovanja računa se pomoću izraza:

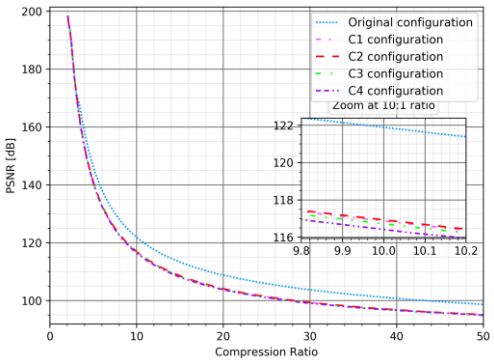
$$\Delta T = \frac{T_{testirano} - T_{referentno}}{T_{referentno}} * 100\%$$

gde je $T_{referentno}$ vreme kodovanja pomoću referentne verzije, a $T_{testirano}$ je vreme kodovanja pomoću konfiguracije čiju složenost želimo da uporedimo sa referentnom konfiguracijom [1]. Za $T_{testirano}$ uzimamo 4 različite konfiguracije(C1, C2, C3,C4), dok za $T_{referentno}$ uzimamo float vrednosti koje su izračunate direktno iz formule $[F] = \frac{2}{N}[A_N][f]$ koja predstavlja DCT transformaciju u matričnom obliku. U okviru tabele 2 možemo videti prosečne vrednosti rezultata konfiguracija za jednu seizmičku sliku.

TABELA 2 Rezultati konfiguracije za jednu sliku 1

| Configuration | Encoding Time (EncTime) | Decoding Time (DecTime) | Average BD-PSNR [dB] |
|---|---|---|---|
| C1 | -12.55769961 | -0.47514177919 | -1.8620 |
| C2 | -14.30100888 | -0.29091856140 | -1.9466 |
| C3 | -14.63641928 | -1.1960734204 | -1.8642 |
| C4 | -14.9005203 | -0.85439783128 | -2.5116 |

Pošto su razlike u parametrima konfiguracije relativno male, što se može videti u tabeli 1, to se takođe reflektovalo i na razlike u rezultatima datih konfiguracija. U okviru EncTime C1 daje najlošiji rezultat ubrzanja od -12.55769961 dok C4 daje najbolji rezultat ubrzanja od -14.9005203. U okviru DecTime C3 daje najbolji rezultat ubrzanja -1.1960734204 dok C2_ daje najlošiji rezultat ubrzanja od -0.29091856140. Što se tiče gubitka u BD-PSNR najbolji rezultat je onaj koji je što bliže 0, međutim plava kriva na slici 13 predstavlja originalnu konfiguraciju i od nje nema bolje. C1 od -1.8620 ima najmanja odstupanja od originalne krive dok C4 od -2.5116 ima najveća odstupanja od originalne krive.




Slika 13 Prosečne „rate-distortion“ (RD) krive za različite konfiguracije kodera analizirane u tabeli 2.

Imajući u vidu EncodingTime, DecodingTime i PSNR možemo reći da su rezultati za C3 konfiguraciju najbolji i ona predstavlja traženi optimum. Cilj ovog testiranja nije bio dobiti što bolji PSNR već pokazati funkcionalnost predloženog kod generatora, kroz mogućnost testiranja 4 različite konfiguracije sa 4 različite preciznosti.

## 10. ZAKLJUČAK

U radu je opisan generator koda koji može da se koristi kao proširenje funkcionalnosti postojećih implementacija binDCT računskog algoritma. Za potrebe eksperimentalne potvrde rezultata kompresije koji se postižu korišćenjem konstruisanog generatora koda korišćen je primer kompresije seizmičkih slika, [2]. Zadatak predloženog kod generatora je bio da na osnovu vrednosti parametra $p$ i $u$ elemenata automatski napiše potreban izvorni kod u programskom jeziku C++, odnosno generiše .cpp i .h fajlove sa potrebnom strukturom i sadržajem, a koji će se koristiti za potrebna izračunavanja u okviru binDCT. Time je omogućeno da parametri binDCT budu promenljivi, što utiče na rezultate kompresije i omogućava izbor optimalne implementacije. Generisani fajlovi se bez dodatnih podešavanja mogu direktno koristiti u standardnim implementacijama algoritama za kompresiju koji koriste binDCT, kao što je npr. [2]. Analizom eksperimentalnih rezultata određeno je koja konfiguracija (generisani kod) daje najbolje rezultate, a koja ima velika odstupanja. Na taj način moguće je odrediti optimalnu konfiguraciju, što u ranijim implementacijama binDCT nije bio slučaj.

## 11. LITERATURA


[1] M. Radosavljević, Z. Xiong, L. Lu, D. Hohl, and D.Vukobratović, “High bit-depth image compression with application to seismic data,” Proc. VCIP’16, China, 2016.

[2] J. Liang and T. D. Tran, ‘‘Fast multiplierless approximations of the DCT with the lifting scheme,’’ IEEE Trans. Signal Process., vol. 49, no. 12, pp. 3032–3044, Dec. 2001.

[3] W.-H. Chen, C. Smith and S. Fralick, "A fast computational algorithm for the discrete cosine transform", IEEE Trans. Commun., vol. 25, no. 9, pp. 1004-1009, Sep. 1977.

[4] M. Radosavljević, A novel algorithm for high bit-depth seismic data compression, PhD thesis, Faculty of Technical Sciences, Novi Sad, 2021.


### Kratka biografija:


**Miloš Marković** rođen je u Rumi 1993. Studije Energetike, elektronike i telekomunikacija, smer Obrada signala, uspešno je završio 2021.god. i stekao uslov za odbranu master rada. Kontakt: markovicnmilos@gmail.com