Prednáška 12

Toto si len pozrieť – nič konkrétne

Adaptivita pri kompresii a prenose obrazu a videa

Motivácia

- prispôsobenie sa na špecifické vlastnosti prenosovej cesty
- prispôsobenie sa na schopnosti zobrazovacieho zariadenia
- preferenciami a požiadavkami konzumenta obsahu
- ...

Cieľ

- spravidla maximalizácia koncovej kvality obsahu vzhľadom na možnosti prispôsobenia sa.
- Napr. z pohľadu
  - prevádzkovateľa prenosovej cesty - maximálna kvality prenášaného obsahu pri stanovenej záťaži siete, resp. jej jednotlivých častí.
    - Môže sa aj jednať o optimalizáciu (prispôsobenie sa) pri meniacich sa podmienok prenosovej cesty: mení sa chybovosť, mení sa latencia, a podobne.
    - Alebo sa mení samotná prenosová cesta (a jej poskytovateľ) v pozadí - prepnutie medzi mobilným pripojením, wifi a pevných pripojením.
  - správcu obsahu sa môže jednať o
    - minimalizáciu veľkosti úložiska.
- Z pohľadu užívateľa
  - maximalizácia kvality obsahu určeného jemu a prispôsobeného vzhľadom na jeho osobné požiadavky a zariadenie, ktoré používa.

Aby ste tušili, že či sú nejaké možnosti adaptácie a aké sú.

Možnosti adaptácie v JPEG

Už ten špecifikoval 2 módy vhodné na využitie v procese adaptácie obsahu:

- progresívny mód ([JPEG] v dodatku G): informácie o obraze sú kódované postupne a to tak, že po prijatí časti informácií tieto predstavujú celý obraz, ale s menej detailami, po prijatí ďalších dávok informácií je obraz postupne vylepšovaný. Pritom platí, že najdôležitejšie informácie, t.j. tie, ktoré najviac zmenšujú chybu rekonštrukcie sú presúvané na začiatku a postupuje sa k menej dôležitým informáciám. Toto sa môže diať dvomi spôsobmi
  - volením koeficientov (spectral selection) - spektrálne koeficientysú prenášané od dôležitejších k menej dôležitým
  - postupnou aproximáciou (succesive approximation) - hodnoty koeficientov sa prenášajú od najdôležitejších bitových rovín po najmenej dôležité.
- Hierarchický mód ([JPEG] v dodatku J) umožňuje lepšie prispôsobenie rozlíšeniu zobrazovacieho zariadenia. Obrázok je prenášaný tak, že najprv sú prenesené informácie potrebné pre rekonštrukciu zmenšenej verzie obrazu, potom postupný zvyšky informácií, ktoré umožnia obrázok kvalitne zobraziť postupne pri väčšom a väčšom rozlíšení.

K JPEG2000 –čo zaviedol (2 veci) + príklad (asi to audio) heslovite, MPEG-2-stručne MPEG-4-čo prináša, AVC H.264-1-2 vety

JPEG2000

- zaviedol paketizáciu
- zaviedol koncept vstiev kvality ("quality layers")

Audio

- adaptivita obsahu napr. v telekomunikačných systémoch. Napr. AMR (adaptive multi rate) kodek kóduje 20ms úseky rečového signálu, reprezentujúce 160 vzoriek signálu a v závislosti od aktuálnych podmienoch v mobilnej sieti sa volí verzia údajov s primerane silným kódovaním.
  - pri dobrých prenosových podmienkach sa volí verzia, ktorá umožní preniesť maximum užitočných informácií avšak slabo chránených proti chybám, pri zlých podmienkach sa zvolí verzia prenášajúca málo užitočných informácií, avšak oveľa oodolnejšia voči prípadným chybám.

MPEG-2

- Zaviedol pojem škálovateľnosti pri videu
- obrazový stream je kódovaný v dvoch vrstvách
  - v prvej, základnej vrstve je prenášaný základ, ktorý môže byť ďalej vylepšený
  - doplnkovou vrstvou, ktorá obsahuje dodatkové informácie k dosiahnutiu plnej kvality videa.
- Základné metódy škálovania
  - škálovanie na základe SNR (Signal to Noise Ratio): doplnková vrstvy posiela informácie, ktoré vylepšujú kvalitu videa pri danom rozlíšení

MPEG-4

- prináša škálovateľnosť audia vo forme MPEG-4 SLS (Scalable Lossless Coding)
  - umožnuje kódovať zvyškovú informáciu ako ďalšiu vrstvu a dosiahnuť až kvalitu bezstratového kódovania.
- rozširuje škálovateľnosť kódovania obrazovej informácie MPEG-2 tak, že umožnuje využiť viac ako 1 doplnkovú vrstivu.

MPEG-4 časť 10 /AVC (Advancedvideo coding)/H.264

- naviac okrem priestorového a SNR škalovania zavádza aj časové škálovanie
- Časové škálovanie je, že v čase existuje viacero sekvencií snímkov a teda v prípade potreby je možné doplnkové sekvencie zahadzovať.
  - Toto má za následok redukciu snímkovú frekvencie
- V dodatku G štandardu MPEG-4 AVC po názvom SVC (Scalable Video Coding) sú uvedené doplnkové odporúčania ako škálovateľnosť efektívne využívať.

MPEG-DASH-Nová vec o tom vedieť viac! Môže byť samostatná otázka za viac bodov

MPEG-DASH

- MPEG-DASH (Dynamic Adaptive Streaming over HTTP),
- štandardizovaný v roku 2012
- popisuje interakciu klienta zo servrom pri adaptívnom doručovaní obsahu cez HTTP protokol
- MPEG-DASH je agnostický od kodeku - môže sa použiť s ľubovoľným moderným kódovacím postupom na kódovanie videa, ako napr. H.264, H.265, VP9 atď.
- Princíp
  - na strane servera je obsah rozdelený na segmenty a každý segment je zakódovaný pri rôznych bitových náročnostiach, resp. rôznym sposobom optimalizovaný obsah.
  - Informácia o segmentoch a ich verziách je dostupná prostredníctvom t.z.v. MPD (Media Presentation Descriptor) súboru.
  - Klientské zariadenie si najprv stiahne MPD súbor, na základe toho začne sťahovať video, pričom v závislosti od aktuálnych podmienok na prenosovom kanáli môže optimalizovať akú verziu stiahne pre ďalšie segmenty videa.
  - Samozrejme kritérium je aby stihol včas stiahnuť čo najkvalitnejšiu resp. najvhodnejšiu verziu pre konzumenta.
- MPEG-DASH používajú napr. Netflix a Youtube pri distribúcii obsahu aj pri streamovaní priamych prenosov.
- Existujú systémy, ktoré vedia efektívne využiť škálovateľnosť MPEG-4 SVC pri použití MPEG-DASH
- V súvislosti s MPEG-DASH vyvstáva otázka, prečo klient, keď sú dobré podmienky nestiahne celý stream k sebe, aby keď sa zhoršia podmienky prenosu, nemusel zhoršovať kvalitu? Odpovede sú viaceré, napr.:
  - obmedzenie dostupných prostriedkov klienta (pamäť)
  - načo sťahovať dáta, ktoré si užívateľ možno ani nepozrie
  - prečo zbytočne zaťažovať prostriedky siete v čase, keď ich možno urgentnejšie potrebujú iní
- MPEG-DASH teda umožnuje riadiť doručovanie obsahu pri rôznych podmienkach prenosového kanálu a optimalizovať kvalitu a prenesený objem dát resp. iné aj kritéria (napr. nepriamo výdrž zariadenia na batériu) z pohľadu jednotlivých užívateľov.
- Optimalizácii rozhodovania v klientovi (ktorú štandard MPEG-DASH) nepredpisuje je v súčastnosti venovaného veľa úsilia, napr. [DASH-LOGIKA]

DASH-VR – 2 vety

DASH-VR

- V MPEG-DASH v súčastnosti prebieha výskum a štandardizačné úsilie ako rozšíriť adaptivitu pre doručovanie VR obsahu (DASH-VR). Predpokladajú sa 3 úrovne:
  - základná úroveň - doručované celé 360°video
  - SRD (spatial relationship description) - poskytuje techniku delenia na oblasti (tiles) aby užívateľ dostával len ten obsah na ktorý sa aktuálne pozerá (predstavuje druhý dodatok MPEG-DASH)
  - SRD kombinované so SVHC - pri kombinácii SRD so škálovateľným rozšírením HEVC sa pri delení videa na priestorové oblasti dá efektívne škálovať kvalita pre jednotlivé oblasti.

5G adaptivita-vedieť princíp

Siete 5G a adaptivita

- Pri komunikačných sieťach 5. generácie je dôraz nielen na rastúcu kapacitu siete, ale aj na to, rastúcou inteligenciou siete túto kapacitu ďalej zvyšovať
- boli navrhnuté viaceré rozšírenia siete, ktoré zavádzajú oproti MPEG-DASH aj inteligenciu na strane siete, napr. použitím vyrovnávacej pamäte (cache), a to čo najbližsie k užívateľom
  - ETSI tento komponent nazýva MEC (mobile edge computing) - typicky sa nachádza na okrajie mobilnej siete, t.j. napr. v NODE-B.
  - Takto sa dá zmenšit záťaž zbytku siete, keďže viac záťažená bude iba časť prístupovej siete medzi cache a koncovými užívateľmi.
- Na realizáciu adaptácie na strane siete sa dá s výhodou použiť architektúra SDN/NFV sieťach (Software-defined Networking / network function virtualization) a v moduloch VNF (virtualised network functions), ktoré môžu byť v pozícii EC (edge computing)
- Poskytovaním výpočtového výkonu na hranici siete sa dá efektívne budovať adaptivita na strane siete v sieťach 5 generácie.

Na poslednom cvičení sme si slušali tieto metódy-treba vedieť

Metóda jednoduchej diferencie [IJCSA2014][PIC2008]

- Založené na jasovej zložke
- aktuálny obrázok je odčítaný od predchádzajúceho
- pre každý pixel je odhadnuté či patrí pozadiu. Pixel $obr_n(x, y)$ pozadiu nepatrí ak

$$|obr_n(x, y) - obr_{n-1}(x, y)| > T_h$$

- T_h je stanovená prahová hodnota

Pozitíva	Negatíva
• nízke výpočtové nároky • nízke pamäťové nároky • extrémne rýchlo sa adaptuje na pozadie (však je aj model pozadia mimoriadne povrchný, je to iba predchádzajúca snímka)	• citlivý na šum • vnútro väčších objektov je často interpretované ako pozadie • vělmi citlivé na treshold

Pozadie ako priemer, resp. medián [PIC2008]

Bez selektivity

- pozadie je tvorené ako priemer resp. medián predchádzajúcich K obrázkov
- veľká spotreba pamäti (K obrázkov v pamäti)
- určuje sa pre každý pixel zvlášť (žiadna priestorová korelácia medzi susediacimi pixelmi)

So selektivitou

- každý pixel aktuálneho obrázku je klasifikovaný ako v popredí, alebo ako súčasť pozadia.
- Iba pixely, ktoré sú súčasťou pozadia sa použijú na aktualizáciu modelu pozadia
  - model pozadia je tvorený ako priemer resp. medián pixelov pozadia z predchádzajúcich K obrázkov
  - veľká spotreba pamäti (K obrázkov v pamäti)
  - určuje sa pre každý pixel zvlášť (žiadna priestorová korelácia medzi susediacimi pixelmi)

Pozadie pomocou IIR filtra [PIC2008]

Bez selektivity:

- B_n - je model pozadia pri snímke n
- Aktualizácia modelu pozadia
  - $B_n(x,y) = \alpha\ obr_n(x,y) + (1-\alpha)B_{n-1}(x,y)$
- α určuje rýchlosť učenia, typicky 0.05

So selektivitou

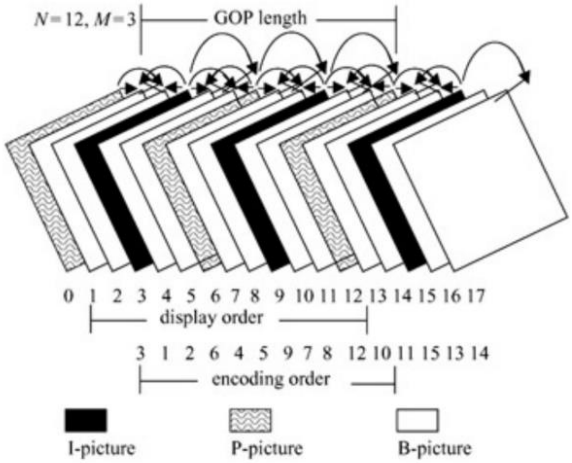
- $B_n(x,y) = \begin{cases} \alpha\ obr_n(x,y) + (1-\alpha)B_{n-1}(x,y) & ak\ obr_n(x,y)\quad patrí\ pozadiu \\ B_{n-1}(x,y) & ak\ obr_n(x,y) nepatrí\ pozadiu \end{cases}$

Prednáška 11

Na 11 prednáške základné princípy H.261,H.263,MPEG-1,MPEG-2,MPEG-4 popravde ešte netuším čo konkrétne. Predpokladám že po 1 vete ku každému to si prosím pozrite v prednáške

Tento obrázok netreba len máme vedieť aké sú rozdiely medzi poradím kódovania a aké sú tam potom závislosti. + môže byť nejaká kontrolná otázka na terminológiu k tejto problematike, ale naozaj neviem čo konkrétne (napr. že čo je to bod). Vraj sme to už preberali na inom predmete.

GOP, poradie zobrazovania a kódovania

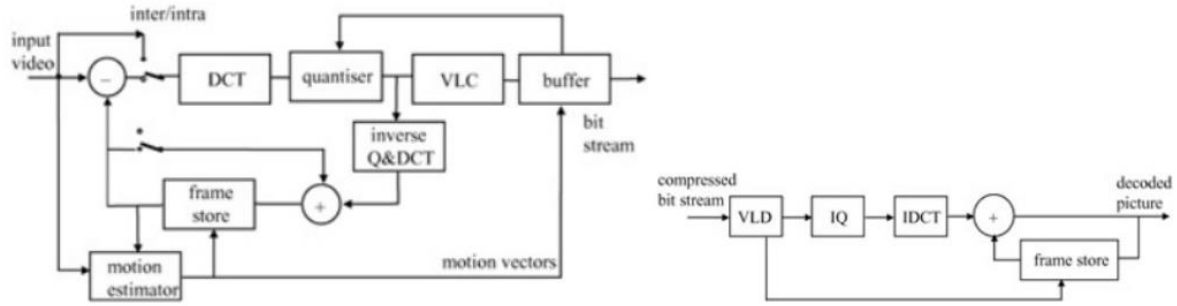


V obrázky GOP(N=12, M=3) je chyba, v strede má byť namiesto I snímky P snímka teda štruktúra je: IBBPBBPBBPBB, IBBPBBPBBPBB, IBBPBBPBBPBB, ...

Vedieť základnú schému z ktorej všetko vychádza

Základné princípy kódovania videaa - schéma generického video kódera [1]

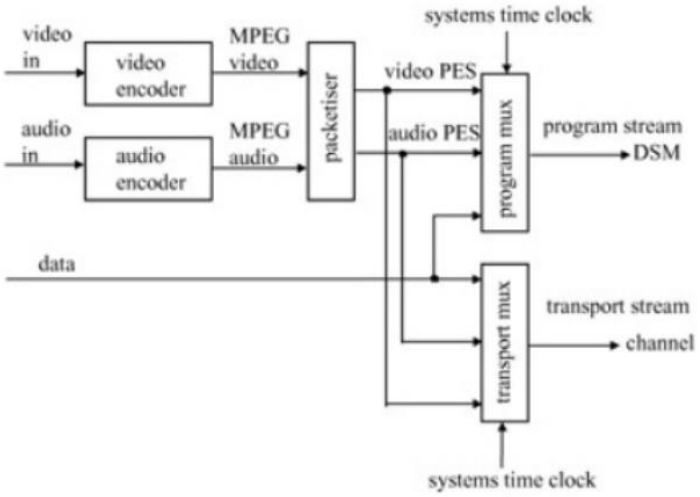
(platí pre H.261, H.263, MPEG-1, MPEG-2, MPEG-4)



MPEG2 – základný princíp + schéma

MPEG2

- MPEG1 bol stavaný pre systémy s malou chybovosťou prenosového kanálu
- MPEG2 zavedený na kódovanie s „vysokou kvalitou“ v r. 1995 na prenos cez B-ISDN pomocou ATM
- Pri prenose cez satelit umožnil prenos 6-8 programov namiesto jedného analógového
- Základný multiplex (vzťahy medzi program/transport streamom a PES (Packetised Elementary Stream)



DSM (digital storage media) – určené pre prostredie bez chýb pri prenose/ukladaní

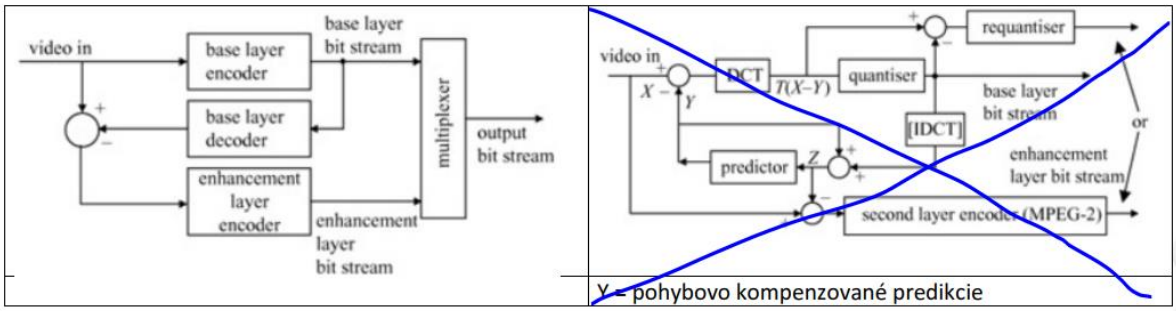
Transportný steam: používa pakety o dĺžke 188 bytov (=4*47bytov, t.j. vhodné ATM bunky s AAL1)

MPEG2 profily – 1-veta – čo sa tam môže zlepšiť +lavá schéma

MPEG2 profily

- Profil: preddefinovaná podmnožina MPEG2 syntaxe (prednastavenia príznakov, ktoré hovoria o prítomnosti elementov, ktoré syntax povoľuje)
- V každom profile sú úrovne (levels), ktoré hovoria o nastavení parametrov, konštánt atď.
- Video stream môže mať viac vrstiev (layers)
  - ak obsahuje iba jednu vrstvu (základnú), je neškalovateľný
  - ak obsahuje naviac ďalšie vrstvy, tie sa volajú vylepšovacie (enhancement)
    - Základná vrstva sa môže chrániť silnejšími korekčnými kódmi
    - Vylepšovacie vrstvy môžu byť chránené menej ...

Princíp viacvrstvového videa (SNR škálovateľnosť):



Y = pohybovo kompenzované predikcie

H.263-1 veta

H.263

- kódovanie pre komunikáciu s nízkou bitovou náročnosťou
- prvá verzia štandardu (ITU-T) je z roku 1995
  - oproti H.261 – polpixelová presnosť pri predikcii pohybu
  - hybridná predikcia
  - odlišné spektrálnych koeficientov a vektorov pohybu (makrobloky stále 16x16)
- Rozšírenia H.263
  - Existoval proces pre H.263+ / H.263++
  - Existoval proces pre H.26L

MPEG4-Pár viet

MPEG4 a obsahovo orientované kódovanie

- Má poskytovať nástroje a rámec pre multimediálne údaje, pre ich
  - ukladanie
  - prenos
  - manipuláciu s nimi
- Je synchrónne s rozšíreniami H.263
- Je to sada odporúčasní
  - Part 1 (ISO/IEC 14496-1): **Systémy**: Popisuje synchronizáciu **a** multiplexing videa a audia (napr. transportný stream)
  - Part 2 (ISO/IEC 14496-2): **Vizuál**: Kompresný formát pre vizuálne dáta (video, nehybné textúry, syntetické obrázky, atď.).
- Používa profily a úrovne
  - Jednoduchý profil

    - Podpora 4 pravouhlých objektov v QCIF (174x144) – Úrovne 1, 2, 3 s prenostmi (64, 128, 384kbit/s)
    - ...
  - Pokročilý jednotuchý profil (Advanced Simple Profile – ASP) , podporuje štvrťpixely, globálny odhad pohybu, B-VOP
  - Hlavný profil – 32 objektov, max. bitrate 38Mbit/s
  - Jednoduchý štúdiový profil ... do 1200Mbit/s ...
  - ...
- Podpora objektov
  - používa VOP (video object planes)
    - Pozadie má vlastnú VOP
    - Jednotlivé objekty v popredí majú vlastné VOP
    -

H.264-Pár viet

H.264 / AVC / MPEG-4 Part 10

AVC=Advanced Video Coding

Porovnanie H.264 s MPEG4-Visual

	MPEG-4 Part2	MPEG-4 Part10
Počet profilov	19	3 (základný, hlavný, rozšírený)
Minimálna veľkosť bloku DCT	8x8	4x4
Presnosť odhadu pohybu	½, ¼ pixel	¼ pixel

Veľa zmien je v detailoch ...

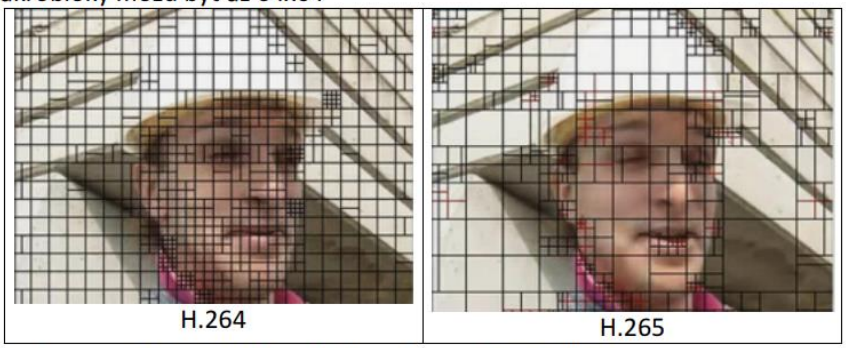
Ako sú kódované obrázky (video picture)?

- Makrobloky sú bloky pixelov obrázku 16x16 (pri jase),, alebo 8x8 pri Cb a Cr
- Každý obrázok sa skladá z N pásov (slice) makroblokov, v každom páse je celočíselný počet makroblokov
- Je 5 typov pásov makroblokov
  - I pás: obsahuje len I makrobloky, t.j. makrobloky, ktoré sú predikované len z predchádzajúcich makroblokov daného pásu
  - P pás: obsahuje len P makrobloky, t.j. makrobloky, ktoré sú kódované z predchádzajúcich zakódovaných obrázkov a/alebo I makroblokov.
  - B pás [základný profil]: B makrobloky, t.j. makrobloky ktoré sú kódované z jedného, alebo dvoch predchádzajúch alebo budúcich zakódovaných obrázkov (1 budúci, 2 budúce, 2 minulé, jeden budúci a jeden minulý, ...), a/alebo I makroblokov.
  - SP pás (Switching P) [rozšírený profil]: obsahuje P a/alebo I makrobloky
  - SI pás (Switching I) [rozšírený profil]: obsahuje SI makrobloky (špeciálne intra makrobloky)

H.265-Pár viet

HEVC, H265, MPEG-H Časť 2

- HEVC (High Efficiency Video Coding) = MPEG-H Part 2 štandard = ITU-T H.265 štandard
- následník H.264/MPEG-4 AVC
  - podporuje rozmery až 8192x4320 (8K UHD)
  - používa veľké CTU (coding tree unit – 16x16, 64x64) – pomáha zvyšovať efektivitu
  - aj makrobloky môžu byť až 64x64



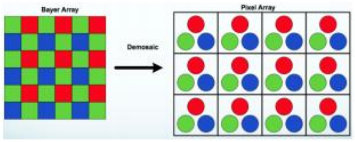
H.264 | H.265

  - 33+2 smerov pre kompenzáciu pohybov (namiesto 9 v AVC)
  - adaptívna predikcia vektorov pohybu
  - používa CABAC ako hlavný entropický kóder (a používa aj Golomb-Rice a Exp. Golombove kódy)
  - zvýšené možnosti paralelizmu (tiles, slices)
  - ako súborový formát sa môže použiť .mp4 (alebo .mkv, .mov, ...)
  - ...

CineForm RAW- 1 veta

CineForm RAW format

- proprietárny CineForm (akvizícia GoPro) vizuálne bezstratový video kodek
- štandardizovaný v.r. 2014 ako Society of Motion Picture and Television Engineers (SMPTE) standard ST 2073 VC-5
- používa waveletovú kompresiu (2/6 wavelet filter)
- 8-24 bitová hĺbka
- RGGB Bayer RAW formát



	Bit-Depth	Native Resolution	Color Space	Compression Algorithm	Compression Ratio	Data-Rate
CineForm RAW™	10-bit	2K	RAW Bayer	VBR Wavelet	5:1 (RAW)	114Mb/s[3]
Sony HDCAM-SR	10-bit	1920x1080	4:2:2 YUV 4:4:4 RGB	MPEG-4 SP	2.7:1 (4:2:2) 4:1 (4:4:4)	440Mb/s
Sony HDCAM	8-bit	1440x1080	3:1:1 YUV[4]	CBR DCT	5.6:1[2]	144Mb/s
Sony XDCAM-HD	8-bit	1440x1080	4:2:0 YUV[4]	VBR MPEG-2	44:1-22:1[2]	18-35Mb/s
Panasonic DVCProHD	8-bit	1280x1080	4:2:2 YUV[4]	CBR DCT	10:1[2]	80Mb/s[3]
HDV	8-bit	1440x1080	4:2:0 YUV[4]	CBR MPEG-2	32:1[2]	25Mb/s

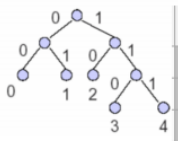
Prednáška 10

Golomb-Rice - teóriu

Golomb - Rice kód

- GR kódy sú optimálne (resp. blízko optimality) pre obojstranné geometrické rozdelenia
- $N = qm + r$, pričom $0 \leq r < m$
  - q je kvocient a r je zvyšok
- kód predstavuje q zakódované unárne (počtom jednotkových bitov) + „0“ + r zakódované ako skrátený binárny kód, nazývaný aj Rice kód.
  - Skrátený binárny kód sa používa pre rovnomerné rozdelenia pravdepodobnosti pre veľkosť abecedy m, pričom m nemusí byť mocninou o základe 2
  - ak $m = 2^r$, potom je výsledok identický s binárnym kódom
  - inak $m = 2^r + b$
    - prvým $2^r - b$ symbolom priradí kódové slová o dĺžke r
    - ostatným $2b$ symbolom priradí posledných $2b$ kódových slov o dĺžke $r + 1$
  - na obrázku sú príklady pre m=5
    - ako vyzerá strom – ukazuje, že výsledný kód je **prefixový**
    - ako vyzerá binárna reprezentácia - šedé hodnoty sú vynechané bity a kombinácie), na reprezentáciu 5 možných zvyškov sa použijú biele bity

m = 5



Kódové slovo		
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Príklad: Golombov kód pre m=5 a náhodne zvolené čísla N

N	q	r	Rice(r)	Golombov kód
2	0	2	10	010
6	1	1	01	1001
9	1	4	111	10111
10	2	0	00	11000
27	5	2	10	11111010

Príklad:
Pre GR kód s m=5 dekódujte z bitov
10000011011011001
=5+0,0+1,5+3,10+1
=5, 1, 8, 11

Run-Length - Teóriu

Run-Length kódovanie (Run Length Encoding – RLE)

- Existuje veľa verzií
- Použitie: kódovanie bitových rovín, čiernobiele obrázky, ...

Základná verzia:

- Na začiatku postupnosti sa predpokladajú nuly a kóduje sa ich počet, následne sa kóduje počet jednotiek, následne núl, atď.
- Počty sa kódujú fixným počtom bitov

Príklad:

Vstupná postupnosť 30 bitov: 0 0 0 1 1 1 1 1 0 1 1 0 0 0 0 0 0 1 1 1 1 1 0 1 1 1 1 1 1 1

- Použijeme 3 bity na kódovanie počtov
  - Počty: 3, 5, 1, 2, 6, 5, 1, 7
  - bity: 011 101 001 010 110 101 001 111
  - Výsledný kód má 24 bitov.
- Použijeme 4 bity na kódovanie počtov
  - Počty sa nezmenia
  - bity: 0011 0101 0001 0010 0110 0101 0001 0111
  - Výsledný kód má 32 bitov
- Použijeme 2 bity na kódovanie počtov
  - Počty: 3, 3, 0, 2, 1, 2, 3, 0, 3, 3, 0, 2, 1, 3, 0, 3, 0, 1
  - bity: 11 11 00 10 01 10 11 00 11 11 00 10 01 11 00 11 00 01
  - Výsledný kód má 36 bitov

Run-Length kódovanie (Run Length Encoding – RLE) 2

- Ako zabezpečíme optimálnu kódovú dĺžku?
- Prečo nepoužiť variabilnú dĺžku?
  - Huffmanov kód
  - Exp. Golombov / Golombov kód
    - zvážime použitie najmä vtedy, ak je potencionálne nekonečná množina symbolov
    - Golombov → vysvetlený na ďalšej strane

Netreba vedieť vzorec pre optimálne m, ale vedieť že existuje optimálne m a prečo existuje a načo sa používa (neviem ešte odpoveď)

Aké m v GR -kóde je optimálne pre RLE verzie 2

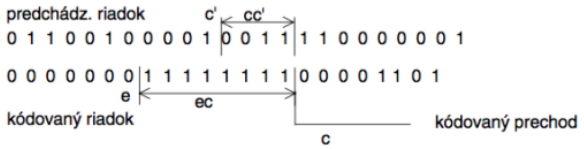
- Nech pravdepodobnosť symbolu 0 je p a pravdepodobnosť symbolu 1 je $1-p$
- Potom optimálne $m = ceil(-1/\log_2 p)$
- Napr. ak $p = \frac{127}{128}$, potom $m = ceil\left(-\frac{1}{\log_2\left(\frac{127}{128}\right)}\right) = ceil(88.38) = 89$

2D- verzia RLE – iba vedieť, že existuje príklad nebude – 1-2 vety stačí

2D verzia RLE – kódovanie bitových rovín

- Používa sa najmä RAC (relative address coding) kódovanie
- Kódujeme napr. po riadkoch
- Môžeme si vybrať čo kódujeme pre každý prechod (príklad: prechod c):
  - Vzdialenosť od posledného prechodu v tom istom riadku (príklad: prechod e, 0->1)
  - Vzdialenosť od rovnakého prechodu (príklad: prechod c', 1->0) v predchádzajúcom riadku smerom vľavo
  - vzdialenosť od rovnakého prechodu v predchádzajúcom riadku smerom vpravo
- Potom potrebujeme
  - Vybrať tú vzdialenosť, ktoré je najkratšia (kódovateľná najmenším počtom bitov)
  - vybrať prefix tejto vzdialenosti, aby bolo jasné o ktorú s uvedených vzdialeností kódujeme

Príklad [Gonzales, Woods, str. 453]:



možná vzdialenosť	vzdialenosť	kód
cc'	0	0
ec alebo cc' (vľavo)	1	100
cc' (vpravo)	1	101
ec	d (d>1)	111h(d)
cc' (c' vľavo)	d (d>1)	1100h(d)
cc' (c' vpravo)	d (d>1)	1101h(d)

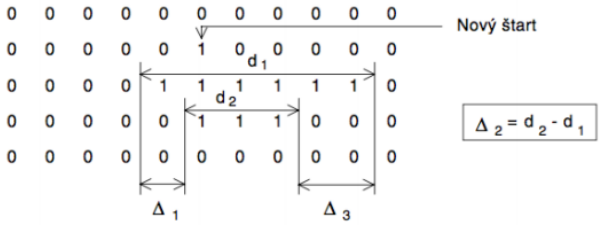
Výsledný kód predstavuje

- prefix možností pre rôzne kombinácie vľavo/vpravo, aktuálny/minulý riadok, d=0, d=1, d>1
- kódovanie vzdialenosti d pomocou vhodného kódu s variabilnou dĺžkou – h(d)

1 – 4	0 ××
5 – 20	10 ××××
21 – 84	110 ××××××
85 – 340	1110 ××××××××
341 – 364	11110 ××××××××××
1365 – 5460	111110 ××××××××××××

Kódovanie bit. Rovín- 1-2 vety, príklad nebude

Kódovanie bitových rovín - kódovanie kontúr objektov: PDQ, DDC



Po štarte kódujeme každý riadok od „štartu“ objektu (jeho „najsevernejší bod“).

1) Najprv kódujeme prvý riadok: kóduje sa pozícia štartu (x,y) a dĺžka prvého riadku d1
2) Potom kódujeme riadky pod
   a. Metóda PDQ (predikčná diferenčná kvantizácia) kóduje pre každý riadok Δ_1, Δ_2, vo vzorcoch $d_2$predstavuje dĺžku aktuálneho a d_1dĺžku predošlého riadku
   b. Metóda DDQ (dvojnásobné delta kódovanie) kóduje pre každý riadokΔ_1, Δ_3
3) Na kódovanie uvedených hodnôt za použije niektorý z kódov s variabilnou dĺžkou slova kódujúci aj znamienko.

Príklad na obrázku:

PDQ	DDQ
1) Kóduj (x,y)=(6,2), d=1 2) Kóduj $\Delta_1 = -1, \Delta_2 = 5$ 3) Kóduj $\Delta_1 = 1, \Delta_2 = -3$	4) Kóduj (x,y)=(6,2), d=1 5) Kóduj $\Delta_1 = -1, \Delta_3 = 4$ 6) Kóduj $\Delta_1 = 1, \Delta_3 = -2$

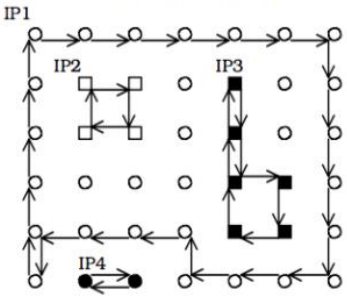
Viacúrovňové kódovanie- 1-2 vety, príklad nebude

Viacúrovňové kódovanie

Podobný spôsob ako je PDQ a DDC kódovanie, je možný realizovať pre kódovanie obrazu, ktorý obsahuje viac kvantizačných úrovní [29]. Principiálne ide o opis uzavretých geometrických útvarov s rovnakou kvantizačnou úrovňou.

Pre realizáciu takéhoto kódovania je potrebné kódovať:
1. úroveň jasu (prípadne poradie farby),
2. štartovací bod príslušného objektu,
3. tvar objektu - uzavretú cestu od štartovacieho bodu opisujúcu obrys objektu a vracajúcu sa späť k štartu.

Postup je už len logickým riešením daného problému. Je potrebné obrysy označovať tak, aby v každom uzavretom reťazci existovala buď len príslušná kvantizačná úroveň alebo ďalší uzavretý objekt, s tou istou podmienkou. Ukážeme si to na príklade so štyrmi úrovňami:



Na záver potrebujeme zakódovať podľa tabuľky znaky napr. v poradí : poradie obrysu (IPx), úroveň, riadková súradnica IPx, stĺpcová súradnica IPx, smer prvého posunu, smer druhého posunu, ...

Kód je samozrejme najúčinnejší pri malom počte kvantizačných úrovní a veľkých jednoúrovňových objektoch.

Kódovanie tvarov – bude iba príklad, preto to sem nedávam

Quad tree kódovanie – 1-2 vety teória

Quad tree úroveň 3

<table><tr><td>b[0]</td><td>b[1]</td></tr><tr><td>b[2]</td><td>b[3]</td></tr></table>	**Vypočítanie indexu pre každý zo 64 blokov:** IndexL3=27b[0] + 9b[1] + 3b[2] + 1b[3] pričom • b[i]=2 ak pixel=255, b[i]=0 ak pixelu=0 • dostávame takto 16 rôznych hodnôt intexu (min=0, max=80)
<table><tr><td></td><td>horný_ index[0]</td><td>horný_ index[0]</td></tr><tr><td>ľavý_ index[0]</td><td>index[0]</td><td>index[1]</td></tr><tr><td>ľavý_ index[0]</td><td>index[2]</td><td>index[3]</td></tr></table>	**Poprehadzovanie indexov (zvýšenie medziblokovej korelácie) v rámci štvorice indexov na základe situácie nad a vľavo od tejto štvorice:** Ak horný_index[0] < horný_index[1], vymeň index[0] s index[1] a index[2] s index[3] okrem sub-blokov 0,1,4,5,16,17,20 a 21 Ak ľavý_index[0] < ľavý_index[1], vymeň index[0] s index[2] a index[1] s index[3] okrem sub-blokov 0,2,8,10,32,34,40 a 42 Ak horný_index[0] + horný_index[1] < ľavý_index[0] + ľavý_index[1], vymeň index[1] s index[2] okrem sub-blokov 0,1,2,4,5,8,10,16,17,20,21,32,34,40 a 42

Quad tree úroveň 2

Analogický obrázok ako pri L3.	**Vypočítanie indexu pre každý zo 16 blokov úrovne 2 zo 4 indexov úrovne 3** Index_L2=27f(indexL3[0]) + 9f(indexL3[1]) + 3f(indexL3[2]) + 1f(indexL3[3]) pričom • f(x)=0 ak x=0 • f(x)=2 ak x=80 • f(x)=1 ináč
Analogický obrázok ako pri L3.	**Poprehadzovanie indexov (zvýšenie medziblokovej korelácie) v rámci štvorice indexov na základe situácie nad a vľavo od tejto štvorice** • Prehodenie je tým istým spôsobom ako pri leveli 3, t.j. vynechávajú sa prvy riadok a stĺpec pri niektorýchvýmenách.

Quad tree úroveň 1

Analogický obrázok ako pri L3.	**Vypočítanie indexu pre každý zo 4 blokov úrovne 1 zo 4 indexov úrovne 2 ako na druhej úrovni**
	Poprehadzovanie indexov nie je.

Quad tree úroveň 0

Analogický obrázok ako pri L3.	**Vypočítanie 1 indexu zo 4 indexov úrovne 2 ako na 1 úrovni**

Kódovanie:

- Máme 85 indexov, ktoré určujú ako čo bolo poprehadzované, tie si musíme pamätať
- Indexy kódujeme pomocou Huffmanovho kódovania(HK) (2 druhy)
- Postupujeme po úrovniach v poradí: 0, 1, 2, 3
- V rámci úrovne postupujeme podľa poradia na obrázku stromu.
- Hodnoty:
  - v úrovni 3 je 64 indexov, každý s 16 možnými hodnotami
    - jeden Huffmanov kód (preddefinovaná tabuľka)
  - v ostatných úrovniach majú indexy 80 možných hodnôt
    - druhý Huffmanov kód (preddefinovaná tabuľka)

Prednáška 9

Aritmetické kódovanie – treba vedieť podrobne + môžu byť mini príklady. Treba vedieť poriadne – je to základ
+ prevod z decimálnej do binárnej sústavy

Nejdem to dávať všetko možno by som si to pozrel v prednáške poriadne

Aritmetické kódovanie (AK) vlastnosti

- Najdôležitejšia vlastnosť: flexibilita
  - Môže sa používať v spojení s ľubovoľným modelom, ktorý produkuje pravdepodobnosť udalostí – dôležité, lebo veľké kompresné pomery sa dajú dosiahnuť len pomocou sofistikovaného modelu vstupných dát
  - Modely môžu byť adaptívne
  - Môže byť použitých viacero rôznych modelov ...
- Druhá najdôležitejšia vlastnosť: optimalita spomenutá na predchádzajúcom slide
  - Najmä ak pravdepodobnosť nejakého symbolu je blízka 1, potom aritmetické kódovanie dáva podstatne lepšie výsledky ako iné metódy.
- Nevýhody
  - Najdôležitejšia = Pomalosť / výpočtová náročnosť
  - Nevhodnosť použitia v aplikáciách v reálnom čase
  - Nutnosť indikovať koniec súboru (buď špeciálnym symbolom, alebo prenášaním počtu bitov výsledného čísla)
  - Slabá odolnosť voči chybám (najmä ak sa používajú adaptívne modely)

Základný algoritmus výpočtu AK

Cieľ:

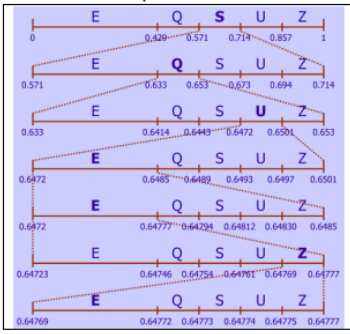
vyslať na výstup jedno desatinné číslo, ktoré presne určuje našu množinu symbolov.
Postupne budeme spresňovať interval, v ktorom sa výsledné číslo nachádza

Vstup:

Majme usporiadanú množinu symbolov a_i ktoré tvoria súbor A o veľkosti N symbolov, ktorý máme zakódovať
Množinu rôznych symbolov a_i označme ako U a jej prvky u_j.
Veľkosť množiny U označme ako M. Pravdepodobnosť každého u_j označme p_j a odpovedajúcu množinu P

Algoritmus:

1) Nastavíme aktuálny interval $[L, H)$, na $[0, 1)$.
2) Opakujeme pre (i=0;i<N;i++)
   a. Aktuálny interval rozdelíme na M podintervalov proporcionálne k p_j
   b. Zvolíme podinterval j , pre ktorý $u_j = a_i$
   c. Na výstup pošleme dostatočný počet bitov, ktoré rozlíšia zvolený interval od všetkých ostatných podintervalov



Napr. kódujme slovo „SQUEEZE“

- Máme 5 rôznych symbolov s pravdepodobnosťami:
  - p(S)=p(Q)=p(U)= p(Z)=1/7=0.143
  - p(E)=3/7=0.429
- výsledok je v intervale 0.64769-0.64772
- t.j. stačí na výstup poslať ľubovoľné čislo z toho intervalu (z čo najmenším počom bitov)
- takéto číslo je $0.101001011101_2 = 0.647705_{10}$
- výsledok je 12 bitov, to je dokonca pod teoretickou hranicou, ktoré je 14.9 bitu (pre vysvetlenie pozri o nasledujúci slide)
- viac informácií nájdete napr. tu: http://www.bilsen.com/aic/cabac.shtml

JPEG2000-1-2vety

Ako je implementovaný bezstratový mód v JPEG2000?

- Požíva sa celočíselný lifting s birtogonálnym waveletom CDF (5,3) (CDF=Cohen-Daubechies-Feauveau)
  - Filtre majú veľkosť 5 a 3
  - Počet nulových momentov DP filtrov je 2 (K-regularita je 2)
- Kvantizačný krok ma veľkosť 1
- Používa sa taktiež EBCOT s artimetickým MQ kóderom
  - kódujú sa všetky bitové roviny

Porovnanie efektivity (približné) – od najhorších k najlepším:

Metóda	Algoritmus		Poznámka
	Predikcia	Entropický kóder	
GIF		LZW	Do 256 farieb
TIFF		LZW	
PNG	Lineárna	LZ77+Huffman	
JPEG	Lineárna	Huffman	Bezstratový mód
FELICS	Max(P1,P2)-min(P1,P2) známe pixely	Golomb-Rice	
JPEG2000	DWT pomocou CDF (5,3)	EBCOT	
JPEG-LS	Mediánová predikcia	Kontextový Golomb-Rice kóder	Metóda LOCO
CALIC	Kontextová lineárna + nelineárna korekcia	Huffman / Aritmetický	m-árny AK s názvom CACM++
FLIF?	MANIAC (Meta-Adaptive Near-zero Arithmetic Coding)		Variant CABACu ...

Poznámky:

- Na porovnanie FLIF verzus VALIC verzus JPEG-LS som zatiaľ nenašiel žiadnu štúdiu ... príležitosť pre študentskú DP?
- Čo že je to ten CABAC ??

H.264-CABAC-1-2vety

Aritmetický kóder v H.264 – CABAC

CABAC=Context Adaptive Binary Arithmetic Coding

- Používa len 2 symboly 0 a 1 (ako binárny AK)
- Využíva, že v H.264 sú predikcie a že pravdepodobnosť symbolu 0 je typicky nad 95%
- Zložitejšie symboly sú „binarizované“
  - Napr. predikčný mód sa binarizuje (8 módov sa prenáša pomocou 3 bitov), každý z týchto 3 bitov sa prenáša ako samostatný symbol
- Používa adaptívne aritmetické kódovanie
  - napr. v predikčnom móde „Coefficient map“ (kódovanie signifikancií)
    - kontext je pozícia pixelu v matici 8x8, t.j. máme 64 kontextov
    - začína s s východzími pravdepodobnosťami, že koeficient má hodnotu 0
    - postupne sa tieto pravdepodobnosti adaptujú aktuálne kódovaným hodnotám
  - predikčný mód „väčší ako 1“ – je veľká pravdepodobnosť, že ak koeficient je nenulový, tak má hodnotu +-1.
  - predikčný mód „absolútna hodnota koeficientu“ – ak koeficient má väčšiu abs. hodnotu ako 1, potom je táto binarizovaná ako exponenciálny Golombov kód. Znamienko nie je kódované v tomto kontexte
  - ...

Golombov kód – vedieť vysvetliť + príklad

Čo je to exponenciálny Golombov kód?

- Je to univerzálny kód
  - Efektívne kóduje malé čísla
  - vieme, aké bity ešte patria ešte akému číslu, priradenie je jednoznačné
- Pravidlá kódovania čísla X do Exp. Golombovho kódu:
  - M=počet bitov čísla (X+1)
  - zakódované číslo v binárnej forme = <M-1 krát bit 0><X+1>

Príklady kódovania čísiel

X_{10}	$(X+1)_2$	M-1	<M-1 krát bit 0><X+1>	H.264 –signed X_{10}
0	1	0	1	0
1	10	1	010	1
2	11	1	011	-1
3	100	2	00100	2
4	101	2	00101	-2
5	110	2	00110	3
6	111	2	00111	-3
7	1000	3	0001000	4
8	1001	3	0001001	-4
...	...	...	...	...

Pri H.264 a H.265 je na niektorých miestach použité aj „predmapovanie“, aby bolo možné kódovať aj záporné čísla

- ak $x \leq 0$ potom sa mapuje na $-2x$
- ak $x > 0$ potom sa mapuje na 2x-1

Príklad: Aké sú toto unsigned čísla ak boli zakódované exp. Golombovým kódom? `00100110110110101100010010010101`
Odpoveď: `3, 0, 0, 2, 2, 1, 0, 0, 8, 4.`

Exp. Golombov kód – pokračovanie

Algoritmus dekódovania (keď nemáme tabuľku):

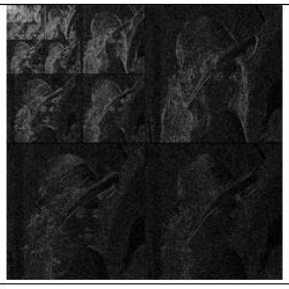
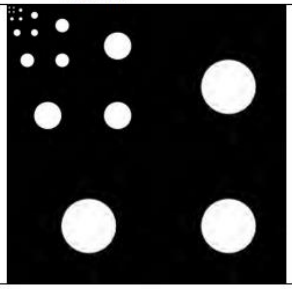
- Spočítaj počet núl po prvú jednotku. K nasledovnej 1 pridaj toľko bitov, koľko bolo na začiatku núl. Odčítaj 1.
- T.j. ak prijatý prúd bitov 00110011001101... aké čísla to znamená?
  - 00 -> k 1 pridám 2 bity -> $110_2=6_{10}$, odpočítam 1 -> prvé číslo je 5
  - 0-> k 1 pridám 1 bit -> 11 -> ..... ->druhé číslo je 2
  - 101 -→ tretie číslo je 4
- znamená to čísla (5,2,4)

Prednáška 8

JPEG2000-vedieť + body 1-6

Ako sa konkrétne kódujú regióny v v JPEG 2000?

- Metóda MAXSHIFT (JP2K part 1)
  - Netreba kódovať hranicu regiónu, lebo:
  - Hodnoty v spektre, ktoré patria nejakému regiónu sa v zmysle bitových rovín jednoducho vysunú o nad ostatné spektrálne hodnoty
    - T.j. najnižšia absolútna hodnota s regiónu bude vačsia ako ľubovoľná hodnota mimo regiónu
  - Takto získajú hodnoty spektrálnych koeficientov "absolútnu prednosť" pri progresívnom prenose informácií
  - Ako ale presne vyzerá "maska", ktorá hovorí aké koeficienty vysunút? Viď príklad masky v (c)

a) Originálny obraz	b) spektrum	c) Ako vyzerá spektrálna maska	d) Rekonštruovaný obraz
			

- Metóda general scaling (JP2K part 2)
  - Polohy regiónov sa pamätajú (používajú sa štvorcové a elipsovité oblasti)
  - Za odmenu môžeme použiť ľubovoľný škálovací faktor a príslušné spektrálne hodnoty môžeme prenásobiť ľubovoľným faktorom dôležitosti

JPEG 2000

Kompresný postup

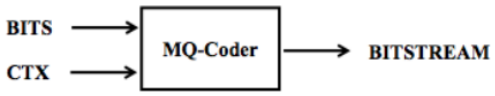
1. Transformácia farebného priestoru
   a. na Y, C_b, C_r
   b. v bezstratovom móde sa uvedený proces iba aproximuje
2. Rozdelenie na diely (tiles) (napr. 128x128,... ktoré sa transformujú úplne nezávisle)
3. Waveletová transformácia
4. Kvantizácia – redukcia počtu bitových rovín
   a. Skalárna kvantizácia s mŕtvou zónou (jednoduché orazanie nižších biových rovín)
      i. JPEG 2000 podporuje oddelené kvantizátory pre každé subpásmo
   b. Trelllisova kvantizácia
5. Po kvantizáci
   a. Každé subpásmo je rozdelené na bloky
   b. Bloky budú kódované nezávisle
6. Entropické kódovanie EBCOT (Embedded Block Coding with Optimised Truncation)
   a. TIER1 - kódovanie zdroja
   b. TIER2 – generovanie výstupného streamu

Truncation = kódovanie každého bloku sa optimalizuje v zmysle bitovej náročnosti verzus skreslenie (rate/distrotion)

EBCOT- vediet čo to je a načo to je takže skôr niekde z Wiki
v prednáške je skôr postup kódovania

Ako funguje EBCOT

- Bitové roviny kódových blokoch sa kódujú pomocou 3 kódovacích prechodov (v uvedenom poradí):
  - Kódovania okolia význačných (significance propagation)
  - Spresnenie magnitúdy význačných (magnitude refinement)
  - Kódovanie nevýznačných (cleanup)
- Začína sa najvyššu bitovou rovinou a postupuje sa k nižším
  - Jednotlivé koeficieny sa pri prekročení hodnoty chategorizujú postupne ako signifikantné (význačné) pričom ich model využíva pravdepodobnosť význačnosti v ich susedstve
- Používajú sa 4 kódovacie postupy (primitívy) kvoli lepšiemu modelovaniu
  - RL – run length
  - ZC – zero coding
  - MR – magnitude refinement
  - SC – sign coding
- V rámci týchto postupov existujú ešte rôzne kontexty
- Informácia o kontexte a jednotlivé bity idú do MQ aritmetického kódera:



Aritmetické kódovanie – poriadne už to spomínal pri 9 prednáške

Aritmetické kódovanie - základ

- entropické kódovanie – používa sa pri stratových aj bezstratových konpresných postupoch
- úloha – zakódovať správu
  - Hufffmanov kód – pristupuje k správe ako k množine symbolov a kóduje každý symbol zvlášť
  - Aritmetický kód – zakóduje celú správu ako jedno jediné číslo
    - Umožnuje sa dostať až k hranici – $\log_2 P$ na symbol

Analýza hraníc efektivity

- Predpokladajme 3 symboly s rovnakými pravdepodobnosťami (je to špeciálny prípad v praxi sa nebude vyskytovať často ☺)
- Vyšlime správu, kde je po 100 výskytov každého symbolu (t.j. dokopy 300 symbolov)
  - Aký je teoretický limit? $pocet_{bitov} = 300(-\log_2 0.\overline{3}) = 300(1.585)$=475.5 [bit]
  - Ako si s tým poradí huffmanov kód?
    - Symbol 1 = 0
    - Symbol 2 = 10
    - Symbol 3 = 11
    - Správa=100+200+200=500 bitov
  - Ako s s tým poradí aritketický kód?
    - Kódujeme číslo v trojkovej sústave, napr.: $0.000 \ldots 000111 \ldots 111222 \ldots 222_3$
    - T.j. napr 100 symbolov, potom 100 symbolov2, potom 100 symbolov3
    - A toto číslo chceme previesť do dvojkovej sústavy
      - $=0.010010 \ldots 01_2$
      - koľko bitov sme spotrebovali, toľko potrebujeme na výsledok
      - na výsledok spotrebujeme približne $300 * \frac{\ln 3}{\ln 2}$=300(1.585)=475.5 [bit]

Prednáška 7

Vedieť Interpolovať

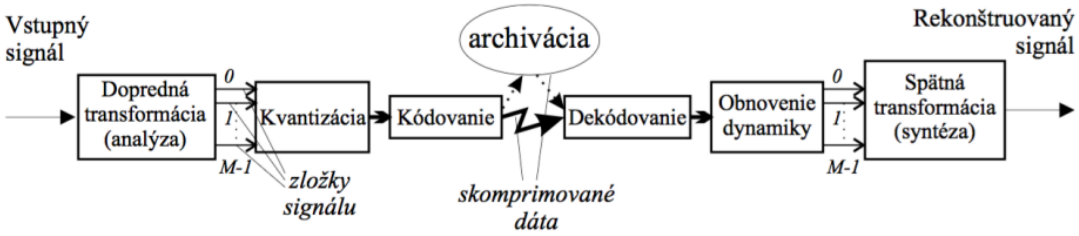
Interpolácia a banky filtrov

- Klasická interpolácia – ako to funguje?
  - Pomocou polynómov
  - Pomocou Splajnov (spline)
    - Polynomy na intervaloch (picewise polynomials)
    - Okrem prepojenia dvoch bodov musí sedieť aj derivácia v koncových bodoch
    - Kvadratické spajny, kubické splajny, …
  - Konštukcia splajnov – aké sú to funkcie?
- Interpolácia pri rekonštrukcii v 2 pásmových bankách filtrov
  - DP Filter $H(z)$ nazývame interpolačný, ak je schopný interpolovať koeficienty prestriedané nulami (najradšej ako polynómy R-tého rádu).
  - Príklady Interpolačných filtrov:
    - Haarov wavelet – konštantná interpolácia h(n)={1,1}
    - CDF 2,2 – lineárna intarpolácia, h(n)={1/2, 1, 1/2},
    - 4-bodová schéma h(n) ={-1/16, 0, 9/16, 1, 9/16, 0, - 1/16}
- V 2D prípade bank filtrov sa dajú zostrojiť separovateľné aj neseparovateľné prípady

Všeobecná schéma kódovania obrazu - vedieť

Ň

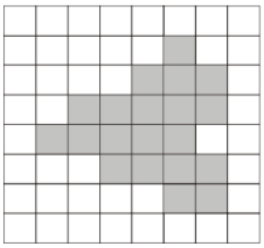
Kódovanie obrazu všeobecná schéma



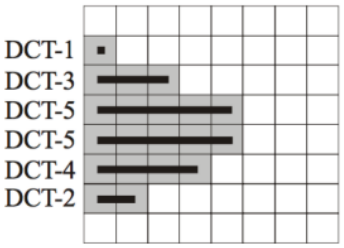
Kódovanie DCT vedieť princíp

Kódovanie objektov

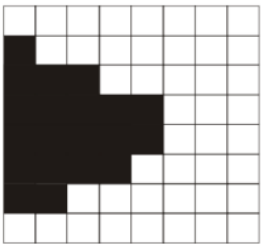
- Klasicky o objekte hovorímeme ak máme jasnú hranicu objektu
- Kódovanie pomocou DWT
  - Pamatáme si hranicu
  - Pamatáme si iba tie koeficienty, ktorých odpovedá bázová funkcia, ktorá sa prekrýva s objektom
- Kódovanie pomocou DCT
  - napr. shape adaptive DCT



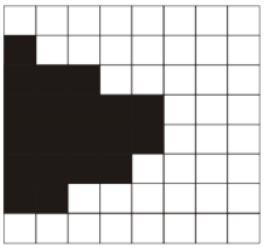
(a) Originálny segment



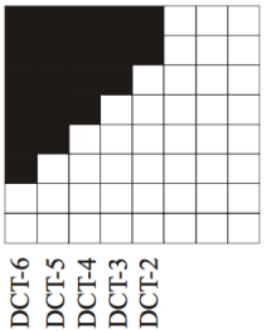
(b) Zoradenie pixelov a horizontálna SA-DCT



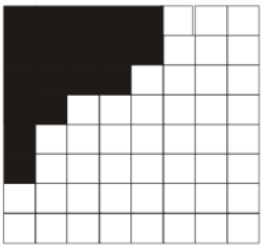
(c) Uloženie 1D koeficientov



(d) Uloženie vzoriek pre vertikálnu SA-DCT



(e) Zoradenie 1D vzoriek a vertikálna SA-DCT



(f) Uloženie 2D SA-DCT koeficientov

SPIHT- pár viet

SPIHT je založený na troch konceptoch:

1. čiastočné zoradenie koeficientov podľa magnitúdy s prenosom pozičnej informácie pomocou algoritmu na triedenie do množín (algoritmus je duplikovaný v dekóderi)

2. postupný prenos zoradených bitových rovín

3. využitie hierarchickej štruktúry spektra DWT

bitová rovina	m_5		m_4		m_3				m_2							
znamienko	s	s	s	s	s	s	s	s	s	s	s	s	s	s	s	s
5	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	→		1	1	0	0	0	0	0	0	0	0	0	0	0	0
3	→		→		1	1	1	1	0	0	0	0	0	0	0	0
2	→		→		→				1	1	1	1	1	1	1	1
1	→		→		→				→							
0	→		→		→				→							

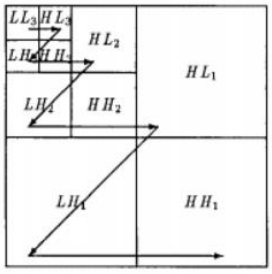
Spresňujúce bity

Obr. 4.9. Príklad koeficientov utriedených podľa magnitúdy

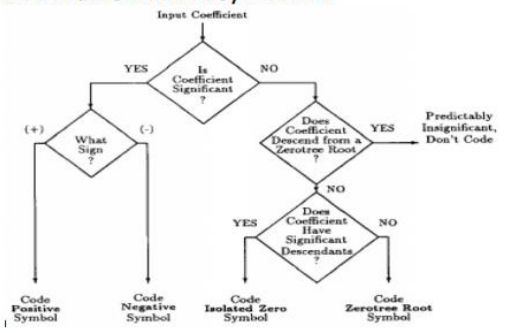
EZW-poriadne + bude aj príklad

EZW – embedded zero tree wavelet (Shapiro 1993)

Poradie skenovania koeficientov



Kódovanie hodnôt symbolmi:



Hlavná slučka enkódera:

$T \leftarrow 2^{\lfloor \log_2(\max(|Y(x,y)|)) \rfloor}$
repeat
  dominantPass(T)
  if image is color **then**
    dominantPass(T)
    dominantPass(T)
  end if
  subordinatePass(T)
  $T \leftarrow T/2$
until $T = 0$

Dominantný prechod

initialize dominant list
while dominant list not empty **do**
  retrieve and remove coefficient from head of dominant list
  if coefficient $\neq$ ZTR **then**
    scan for next coefficient
    code coefficient and add to dominant list
    if coefficient is coded as POS or NEG **then**
      add absolute_value(coefficient) to subordinate list
      set coefficient position in transformed image to 0
    end if
  end if
end while

Podriadený prechod

while not past end of subordinate list **do**
  get next coefficient from subordinate list
  coefficient $\leftarrow$ coefficient $- T$
  if coefficient $\geq T/2$ **then**
    output a '1'
  else
    output a '0'
  end if
end while

Prednáška 6

Lifting – bol na zápočte teraz teória k tomu filting predikcie aktualizácie

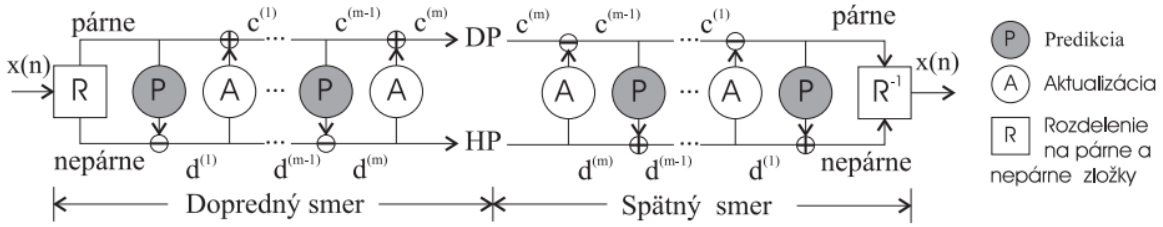
Lifting schéma

- Lifting schéma umožňuje jednoducho zlepšiť špecifické vlastnosti danej WT – odiaľ názov „*lifting*“
- Jednoducho opisuje závislosti medzi pármi filtrov, ktoré zdieľajú ten istý HP, resp. DP filter. Takto môžeme začať z triviálneho prípadu "lenivého" waveletu a postupne vybudovať pár filtrov s požadovanými vlastnosťami, t.j. postupujeme podľa tzv. *Lifting schémy*
- Umožnuje efektívne realizovať klasické WT s nasledovnými výhodami:
  - urýchlenie implementácie WT (napr. v 1D prípade až dvojnásobne)
  - možnosť vykonať výpočty bez použitia prídavnej pamäte (t.j. "in-place")
  - jednoduchý dizajn vlastných WT (naviac so zaručenej invertovateľnosťou)
- Umožnuje jednoducho rozširovať klasickú WT, zaujímavé su napr.
  - konštrukcia nelineárnych WT (napr. adaptívnych, celočíselných)
  - použitie WT pre nerovnomerne navzorkované signály
  - konštrukcia WT na intervaloch, krivkách, povrchoch

Konštrukcia lifting schémy je založená na koncepte polyfázového rozkladu bánk filtrov:

- každú FB, môžme rozložiť (faktorizovať jej filtre) na jednoducho invertovateľnú postupnosť krokov
- Tieto kroky tvoria v algoritme priečkovú štruktúru, ktorú môžme interpretovať ako postupnosť *predikcií* a *aktualizácií* dvoch množín v obraze až po ich vzájomnú dekoreláciu

Principiálne môžeme waveletovú transformáciu implementovanú lifting schémou prekresliť podľa nasledovného obrázku. Sú vyznačené základné bloky lifting schémy: *rozdelenie, predikcia* a *aktualizácia*:



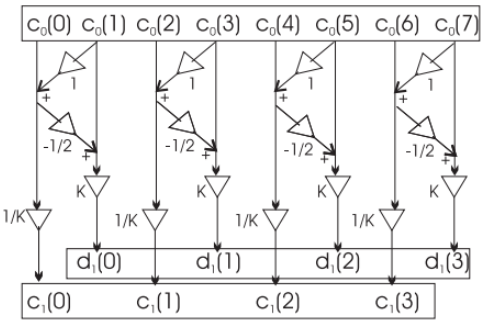
- cieľom predikcie je získanie čo najmenších hodnôt v HP časti po doprednej transformácii.
- aktualizáciou sa snažíme zachovať v DP časti čo najviac vlastností pôvodného obrazu, čo sa stáva dôležitým najmä pri rekurzii v DP časti (aby bola predikcia aj naďalej účinná)

Lenivý wavelet + Haarov vedieť CDF 2,2 (teóriu)

Príklad A: Lenivý wavelet
Aké vlastnosti má naše spektrum?
Vstupný signál je jednoducho podvzorkovaný, takže máme najhorší možný aliasing. Charakter oboch zložiek signálu je rovnaký.

Príklad B: Haarov wavelet
V DP časti chceme dostať priemer susedných koeficientov, v HP časti zase rozdiel. Ak je teda funkcia po častiach konštantná dostávame v HP časti nulové koeficienty.

1) Signál si rozdelíme na párne a nepárne časti:
$$c^{(0)}(n) = x(2n) \qquad d^{(0)}(n) = x(2n+1)$$
2) Aktualizujeme priemer (nenormovaný):
$$c^{(1)}(n) = c^{(0)}(n) + d^{(0)}(n)$$
3) Predikujeme HP časť:
$$d^{(1)}(n) = d^{(0)}(n) - 0.5c^{(0)}(n)$$
4)Normalizujeme ($K = \sqrt{2}$):
$$d(n) = Kd^{(1)}(n) = \frac{\sqrt{2}}{2}(x(2n) - x(2n+1))$$
$$c(n) = \frac{1}{K}c^{(1)}(n) = \frac{\sqrt{2}}{2}(x(2n) + x(2n+1))$$



O prediktoroch - vedieť

O prediktoroch

- Vlastnosti výslednej transformácie sú určené vlastnosťami a spôsobom aplikácie prediktorov
- Teoreticky môžeme použiť ľubovoľné prediktory. Priečková štruktúra nám zaručuje biortogonalitu a tým aj úplnú rekonštrukciu. Môžeme použiť nelineárne prediktory, napr. adaptívne alebo celočíselné. Celočíselné prediktory dostaneme napr. jednoduchým zaokrúhlením hodnoty (nelineárna operácia).

Koncept prediktorov predstavuje silný vzťah medzi *transformačným* a *prediktívnym* kódovaním, kde sa snažíme predpovedať (aproximovať) signál a kódovať iba prípadnú diferencia od originálneho signálu

Lifting schéma takto principiálne umožnuje dvojaký prístup k dekorelácii dát:
a) použiť transformačný princíp (t.j. WT) a celý postup faktorizovať na kroky liftingu
b) b) využiť priamo predikčný princíp a snažiť sa data dekorelovať priamo dizajnom sústavy prediktorov použitých v lifting schéme

Lifting – urýchlenie výpočtu – pár viet

Lifting – aké je urýchlenie vypočtov?

- Výpočet FFT – komplexnosť rádu N*log (N)
- Výpočet DWT – komplexnosť rádu N
- Výpočet DWT liftingom – ďalšie *zníženie počtu operácií až na polovicu*.

Priemerný počet násobení a sčítaní na výpočet jedného koeficientu pri DWT jednostupňovom rozklade:

Typ waveletu	DWT klasicky	DWT liftingom	Urýchlenie
Haar	3	3	0%
Db2	14	9	56%
Db3	22	14	57%
FBI (9,7)	23	14	64%
B-spline(4,2)	17	10	70%
Interpolačné(N,Ntilde)	3(N+Ntilde)-2	3/2(N+Ntilde)	100%

Príklad C: CDF 2,2 wavelet

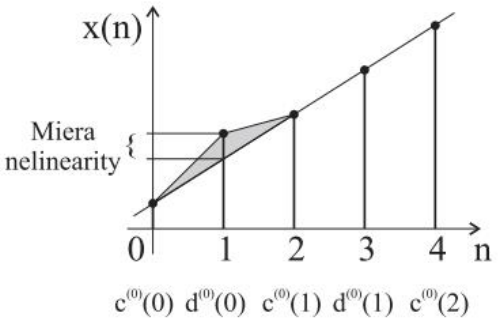
Čo ak chceme reprodukovať po častiach lineárme funkcie?

1) Najprv si rozdelíme signál na párne a nepárne časti:

$$c^{(0)}(n) = x(2n) \qquad d^{(0)}(n) = x(2n+1)$$

2) Waveletové koeficienty nám vtedy určujú mieru, ako sa náš signál líši od lineárneho:

$$d^{(1)}(n) = d^{(0)}(n) - \tfrac{1}{2}\left[c^{(0)}(n) + c^{(0)}(n+1)\right]$$



3) V DP časti chceme zachovať aspoň priemer, t.j. jednosmernú zložku signálu. Hľadáme riešenie v tvare:

$$c^{(1)}(n) = c^{(0)}(n) + A\left[d^{(1)}(n) + d^{(1)}(n-1)\right]$$

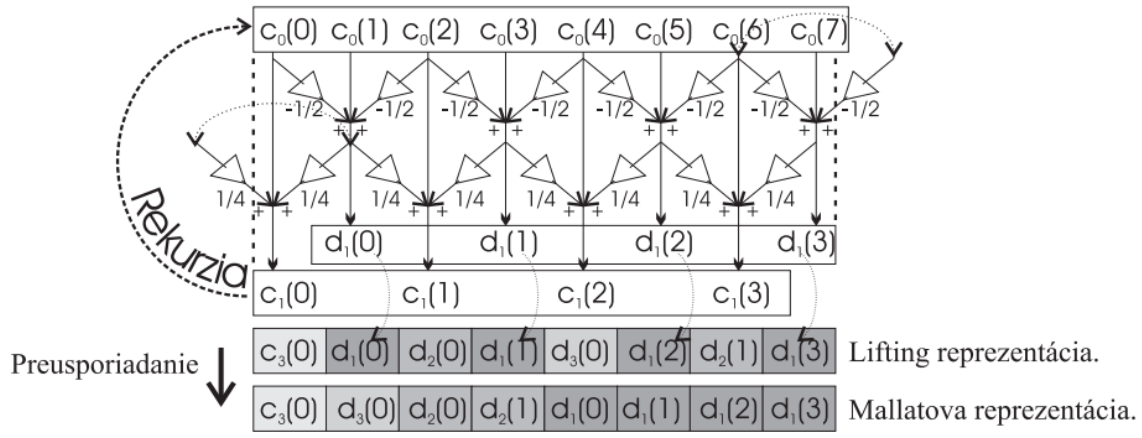
Ak uvážime, že koeficientov c(n) je polovičný počet ako x(n), musí platiť:

$$\sum_n c^{(1)}(n) = \tfrac{1}{2}\sum_n x(n)$$

dostanemie hodnotu A=1/4.

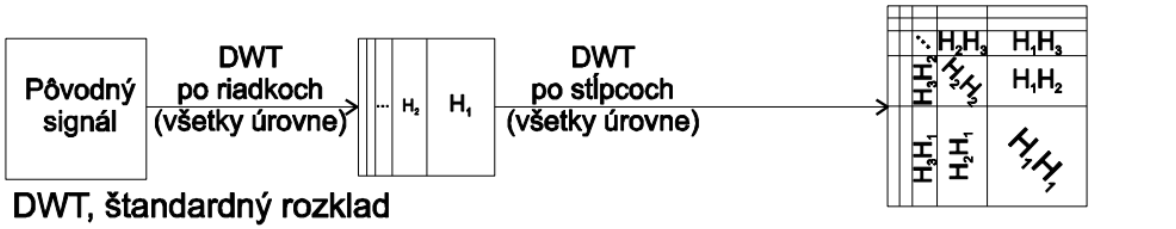
3) Dostali sme CDF 2,2 filter s $\tilde{h}(n) = \left(-\tfrac{1}{8}, \tfrac{1}{4}, \tfrac{3}{4}, \tfrac{1}{4}, -\tfrac{1}{8}\right)$

Reprezentácia signálu po liftingu (príklad CDF 2,2 wavelet)

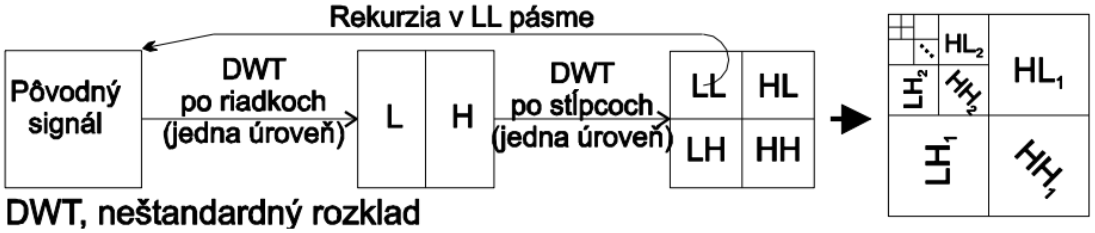


2D DWT – mali by sme to poriadne vedieť – princíp výpočtu

Základné riešenia 2D DWT



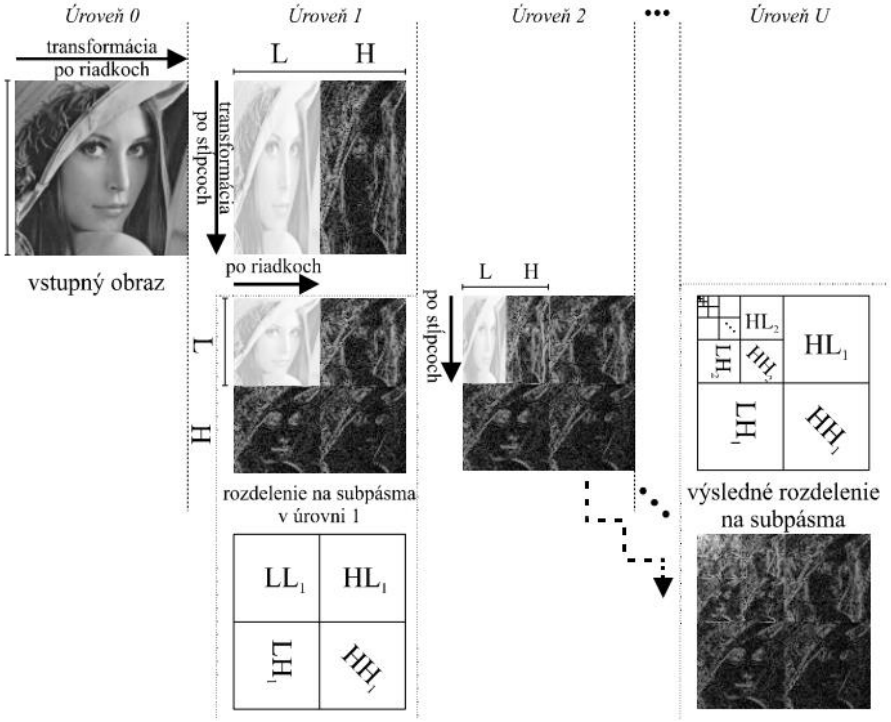
DWT, štandardný rozklad



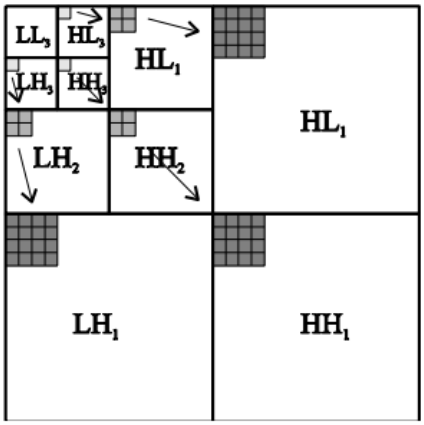
DWT, neštandardný rozklad

Výhody DWT s neštandardným (NS) rozkladom:

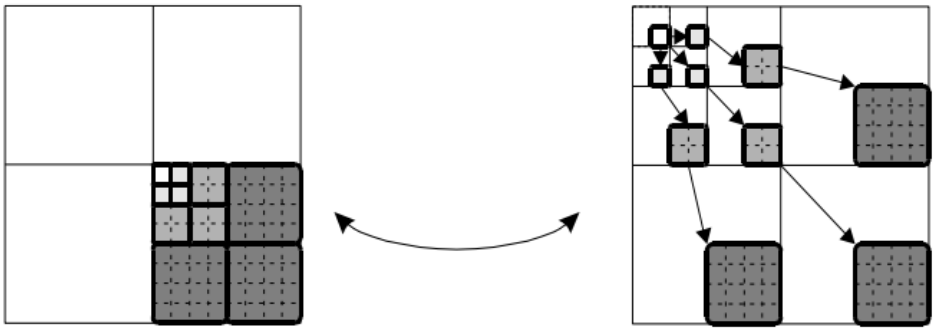
- menší počet potrebných operácií oproti štandardnému rozkladu
- lepšia reprezentácia signálu v priestore.



Spôsob výpočtu 2D DWT rozkladom na subpásma v banke filtrov
(v každom transformačnom kroku sa signál rozdelí na časť s dolnopriepustným L (Low) a hornopriepustným H (high) charakterom, pričom v krokoch sa striedajú smery x *a y)*



Zobrazenie priestorových závislostí v NS DWT spektre 2D signálu



a) V blokoch oddelené spektrá pri blokovej transformácii

b) Hierarchické rozdelenie subpásiem pri 2D DWT

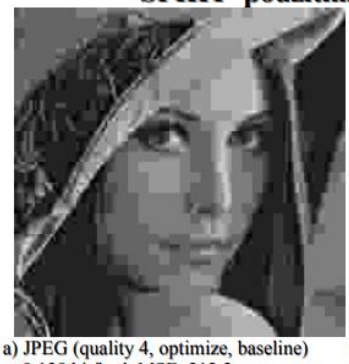
Porovnanie tvaru spektier pri Blokovej transformácii a DWT.

a) Haar , MSE: 454.3 b)Daubechies 4, MSE: 391.24 c) CDF 5/3, MSE: 368.5

d) CDF 5/3 s vymenenými filtrami pre analýzu a syntézu, MSE: 546.5 e) Daubechies 20, MSE: 322.4 f) FBI 9/7, MSE: 324.3

Aproximácia obrazu vybranýcmi waveleovými transformáciami pri vynulovaní *98.5 % waveletových koeficientov*

Porovnanie kompresných výsledkov štandardu JPEG a progresívneho kódovania algoritmom SPIHT použitím DWT a DCT (výrez obrazu Lena, veľkosť 256x256)



a) JPEG (quality 4, optimize, baseline)
0.139 bit/bod, MSE: 212.2

b) DCTII, bloky 8x8, SPIHT,
0.139 bit/bod, MSE: 203.38

c) DCTII, bloky 16x16, SPIHT
0.139 bit/bod, MSE: 134.64

d) DWT Daubechies 20, SPIHT
0.139 bit/bod. MSE: 137.92

e) DWT CDF 9/7, SPIHT
0.139 bit/bod, MSE: 107.3

f) DWT CDF 9/7, SPIHT
0.053 bit/bod, MSE: 241.3

Aplikácie waveletov

Spojité/diskrétne signály všeobecne

- detekcia diskontinuít (aj v deriváciách)
- zistenie charakteru signálu (škálogramy)
- odstraňovanie šumu - kompresia signálov

Audio aplikácie:

- kompresia zvuku waveletovými paketmi
- audio broadcasting
- odstraňovanie šumu
- watermarking

Spracovanie obrazu & videa

- kompresia statického obrazu a sekvencií obrazov -> JPEG 2000, Motion Jpeg 2000
- odstraňovanie šumu
- počítačová grafika
- watermarking

Prenosové systémy

- multiplexery a demultiplexery na báze bánk filtrov

Prednáška 5

DWT na okraji signalu - vedieť

Čo robiť pri DWT na okraji signálu signálu:

Ak má čo chceme?

- Nadbytočnú reprezentáciu?
- Zachovať ortogonalitu ?
- Nemeniť tvar funkcií (t.j. ich vlastnosti)?

Možnosti:

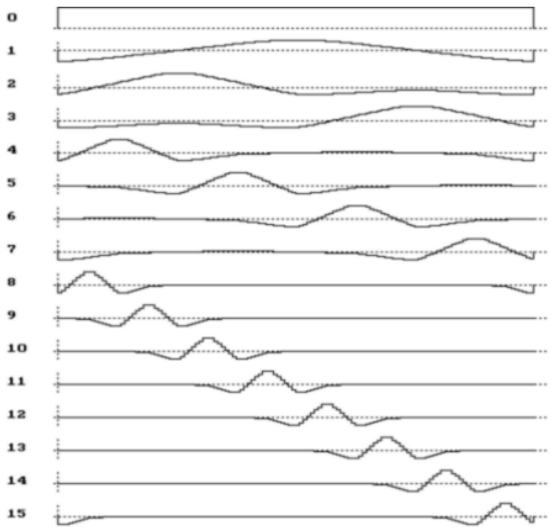
1. *Doplnenie nulami* - vnáša diskontinuity na okrajoch signálu.
2. *Periodifikácia signálu* - má za následok periodifikáciu analýzy s rôznym rozlíšením (MRA), čo je v praxi implementované kruhovou konvolúciou pri filtrovaní v čase. Výsledky sú lepšie ako v 1. prípade .
3. *Symetrické rozšírenie signálu* - je podmienené použitím filtrov s lineárnou fázou , t.j. biortogonálnymi waveletmi.
4. *Priama extrapolácia*. Nepredpokladáme *žiadnu symetriu*, pričom okrajové hodnoty mimo hraníc signálu sa vyjadrujú pomocou transformačných koeficientov (lineárna, polynomická, nelineárna závislosť hraničných hodnôt).

Periodické rozšírenie -vedieť

2

Periodické rozšírenie.

- **speriodifikované transformáčné matice**
- **speriodifikovanie bázových funkcií ("pomalšie funkcie dokonca musia meniť tvar)**



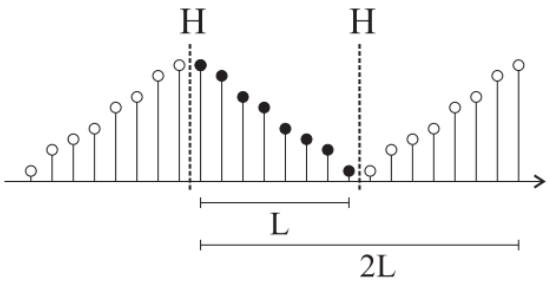
Prvých 32 bázových funkcií diskrétnej waveletovej bázy o veľkosti N=128 pre FBI(9,7) wavelet

Čo sa deje ak N != 2^n ?

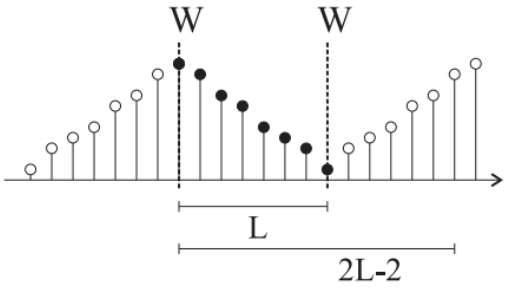
Symetrické rozšírenie - vedieť

Symetrické rozšírenie signálu

I) polbodová symetria(H)



II) celobodová symetria (W)



Pre aké filtre(sekvencie $h_{mr}(n)$,$g_{mr}(n)$) použiť ktorú symetrizáciu?

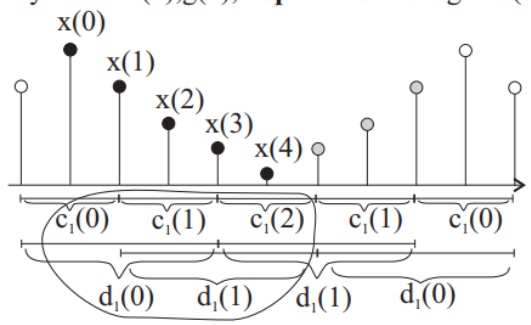
a) $h_{mr}(n)$,$g_{mr}(n)$ majú párnu dĺžku(t.j. sami sú H symetrické) → typ H

b) $h_{mr}(n)$,$g_{mr}(n)$ majú nepárnu dĺžku(t.j. sami sú W symetrické) → typ W

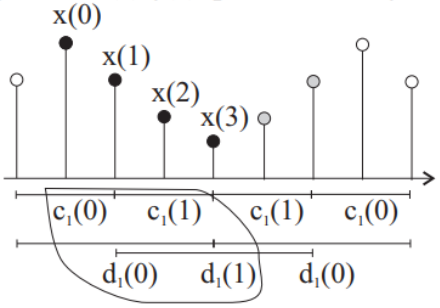
Typ symetrizácie signálu a symetria dilatačných koeficientov musí byť zhodná aby rozšírený signál po podvzorkovaní danú symetriu nestratil.

Stred filtrov a reprezentácia signálu s párnou/nepárnou dĺžkou

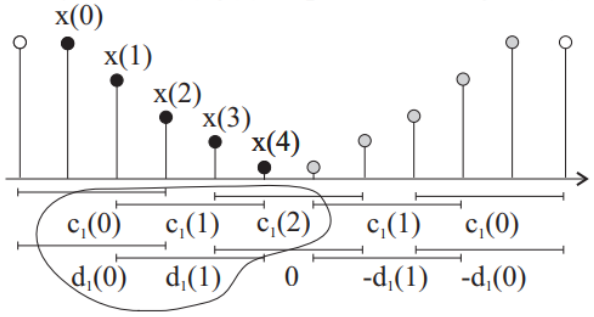
W symetria h(n),g(n), **nepárna** dĺžka signálu(5)



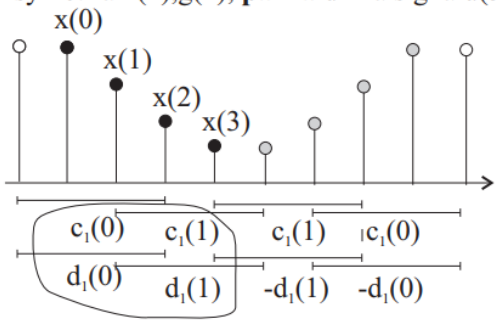
W symetria h(n),g(n), **párna** dĺžka signálu(5)



H symetria h(n),g(n), **nepárna** dĺžka signálu(5)



H symetria h(n),g(n), **párna** dĺžka signálu(5)



Rozšírenia/zovšeobecnenia waveletov:

- wavelety na intervale
- **viacrozmerné wavelety**
- **M-pásmové wavelety**
- Multivavelety
- **waveletové pakety**

Viacrozmerné wavelety – vedieť rozdiel medzi separovateľným a neseparovateľným waveletom

Viacrozmerné wavelety(1)

Triviálne, ***separovateľné prípady*** viacrozmerných waveletov môžeme vytvoriť pomocou tenzorového súčinu použitím 1D prototypov týmito spôsobmi [8]:

- *Štandardný prípad* — tenzorovým súčinom 1D bázových funkcií. Výsledná 2D báza bude teda tvorená súčinmi $\varphi_{i,n}$ a $\psi_{j,n}$, $i, j, n \in \mathcal{Z}$. Napr. pri počiatočnej úrovni rozlíšenia 0 a pri U úrovniach rozkladu bude výsledkom $(U+1)^2$ množín funkcií, pomocou ktorých bude signál reprezentovaný:

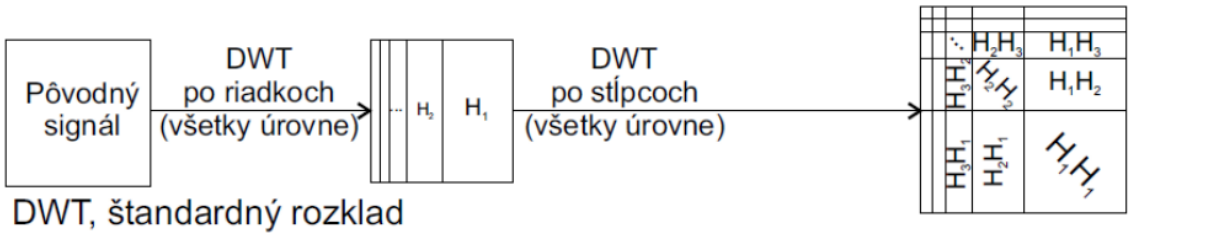
$$\begin{matrix} \varphi_{U,n}(x) \times \varphi_{U,n}(y) & \psi_{j,n}(x) \times \varphi_{1,n}(y) \\ \varphi_{1,n}(x) \times \psi_{i,n}(y) & \psi_{j,n}(x) \times \psi_{i,n}(y) \end{matrix} \qquad n \in \mathcal{Z}, \quad i, j = 1 \ldots U\,. \tag{4.8}$$

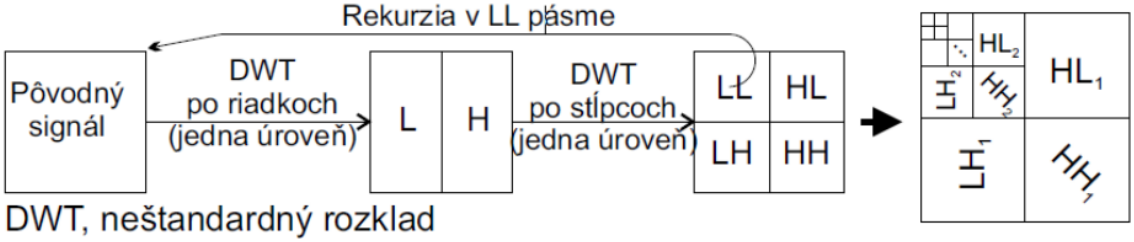
- *Neštandardný prípad* — tenzorovým súčinom analýz s viacúrovňovým rozlíšením (AVR) [18]. V 2D prípade sú potom bázové funkcie tvorené zmenami mierky a posunmi troch základných waveletov $\psi\varphi(x,y)$, $\varphi\psi(x,y)$, $\psi\psi(x,y)$ a funkcie mierky $\varphi\varphi(x,y)$:

$$\varphi\varphi(x,y) = \varphi(x)\varphi(y) \qquad \psi\varphi(x,y) = \psi(x)\varphi(y) \tag{4.9}$$

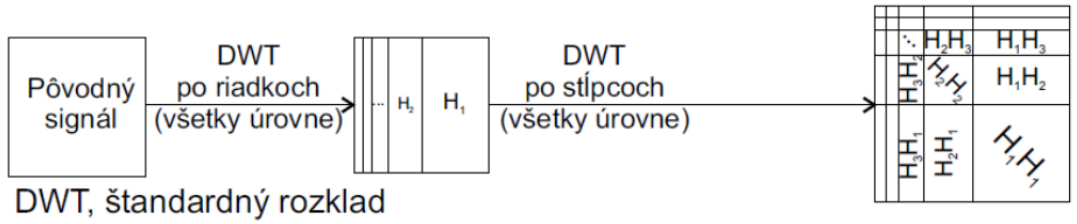
$$\varphi\psi(x,y) = \varphi(x)\psi(y) \qquad \psi\psi(x,y) = \psi(x)\psi(y)\,. \tag{4.10}$$

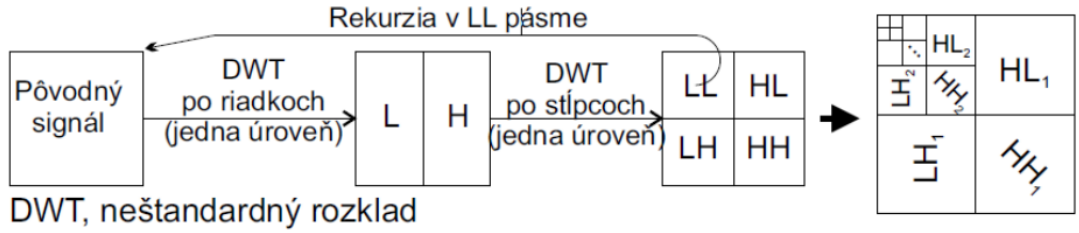
Viacrozmerné wavelety(2)



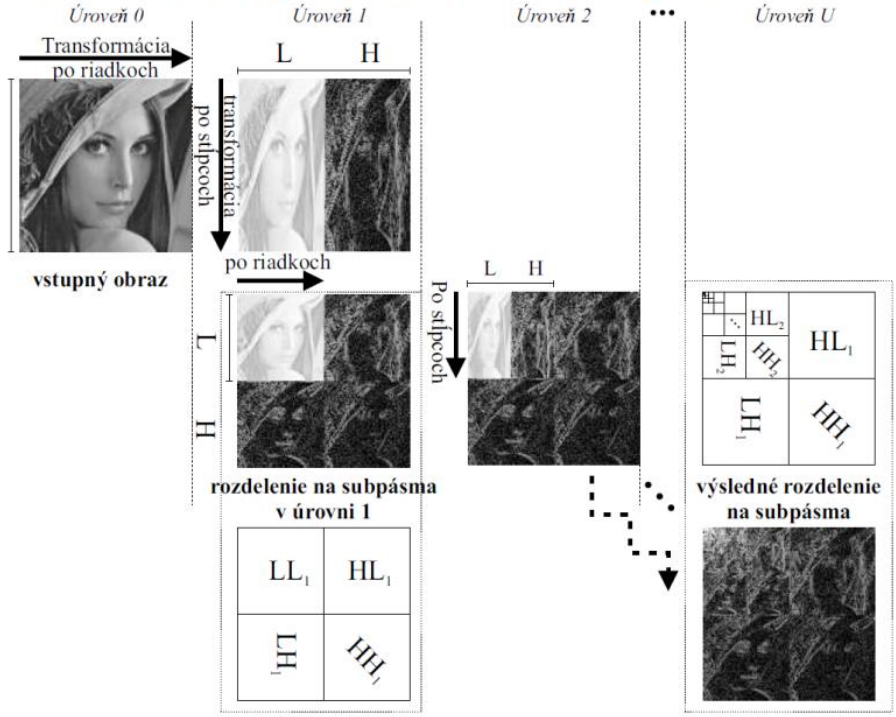


Viacrozmerné wavelety(3)

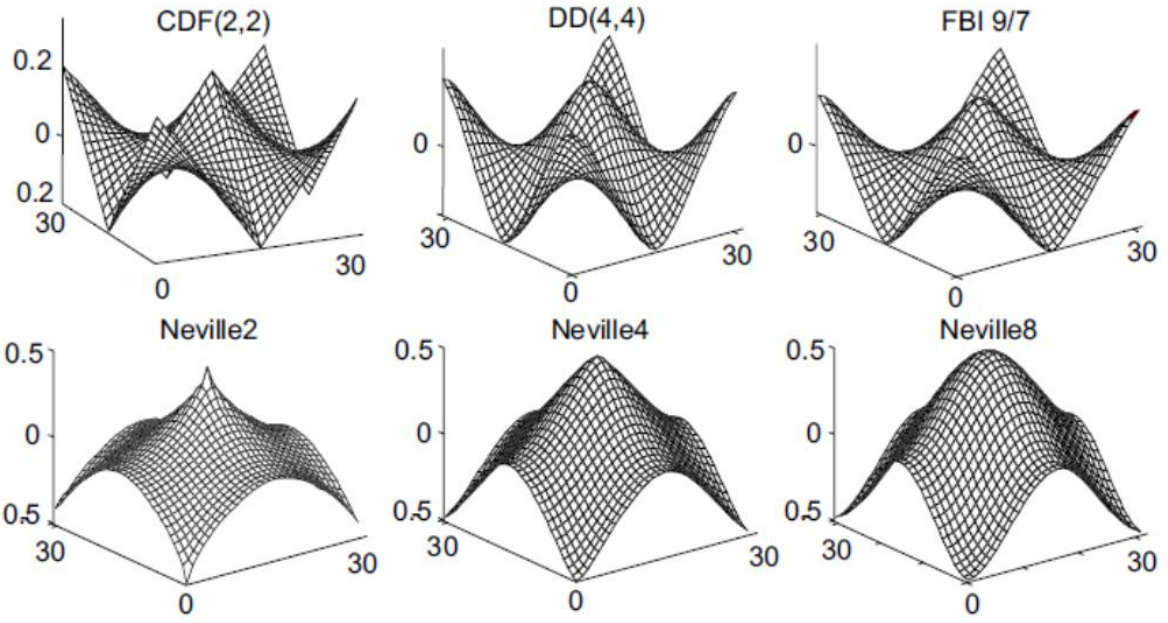




Viacrozmerné wavelety(4) – „neštandardný“ rozklad



Viacrozmerné wavelety(5)
separovatľné a neseparovateľné wavelety



Viacrozmerné wavelety(6)
separovatľné a neseparovateľné wavelety – aprofimácia obrazu

Originál | Aproximácia, FBI 9/7 | Aproximácia, Neville8

Porovnanie aproximačných vlastností separovateľného systému s FBI 9/7 waveletom a neseparovateľného systému s Nevillovými waveletmi 8.rádu. Výsledky sú zobrazené pri zachovaní 1/800 pôvodnej informácie a použití kompresného algoritmu SPIHT.

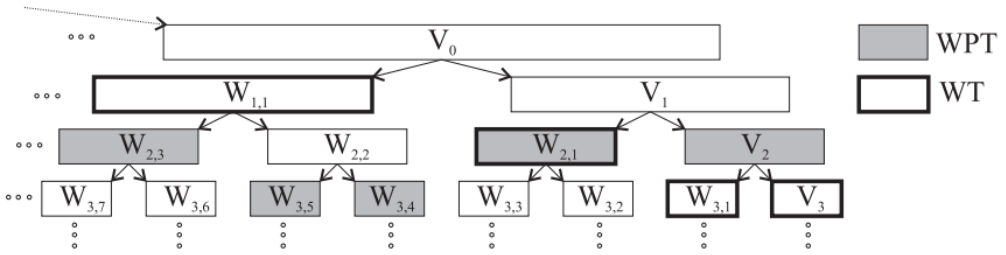
Waveletové pakety – poriadne

17

Waveletové pakety

- v klasickom prípade waveletov rozkladáme v MRA iba sumačné podpriestory V_m
- waveletové pakety rozklad zovšeobecňujú aj na diferenčné priestory W_m
- sú možným rozšírením všetkých doteraz uvedených typov waveletov
- umožňujú adaptívnu, podrobnejšiu a flexibilnejšiu analýzu signálov

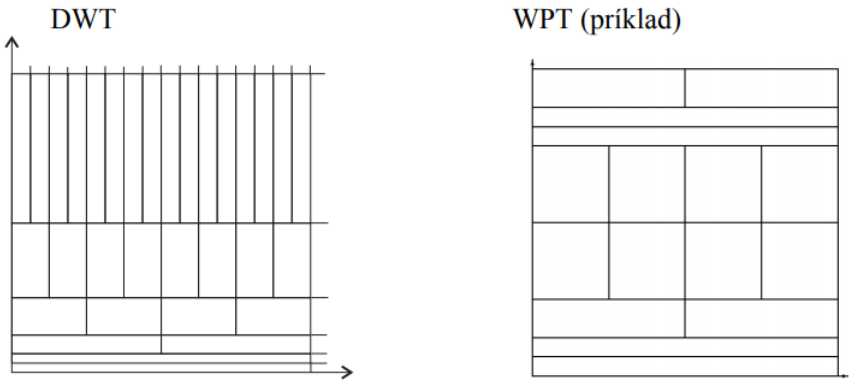
Výpočet Waveletovej paketovej transformácie(WPT) – v MRA je dovolené deliť aj diferenčné popriestory, nielen sumačné:



Vzniká kompletný waveletový paketovým strom. Reprezentácia kompletným stromom je *redundantná* - stačí použiť iba *vhodnú časť stromu*.

18

Delenie priestorov odpovedá aj deleniu časovo-frekvenčnej roviny



WPT umožňuje *adaptívne* resp. optimalizované delenie podpriestorov = delenie časovo-frekvenčnej roviny = použitie istej časti kompletného waveletového paketového stromu.

Ktorú časť stromu použiť – ako čo najlepšie reprezentovať signál ?
Výber najvhodnejšej stromovej štruktúry je ekvivalentný s *hľadaním najlepšej bázy*.
Najbežnejšie kritériá pre výber najlepšej reprezentácie signálu, formované pomocou tzv. *nákladovej funkcie* λ sú:

- minimalizácia entropie reprezentácie signálu (Wickerhauser, Coifman)
- minimalizácia počtu bitov reprezentácie signálu a skreslenia pri danej množine kvantizátorov

Koľko je možných báz?
Ak dĺžka signálu je N, potom pre α , počet možných WPT báz platí:

$$\alpha \geq 2^{N/2}$$

Ako teda vyberať najlepšiu bázu?
Najjednoduchšie je rozhodovať sa priamo počas rozkladov, t.j. rozhodovať sa, či ďalší rozklad prevedieme, alebo nie.

Aké kritérium použiť?
Rozhodujeme sa, podľa toho či *náklady po rozdelení budú väčšie ako pred rozdelením*. Kritériom musí byť taká nákladová funkcia, ktorej aditivita sa rozkladom pri DWT zachováva (napríklad entropia).

Príkladom je napr. Shannonova entrópia E zo signálu s(n), pre ktorú

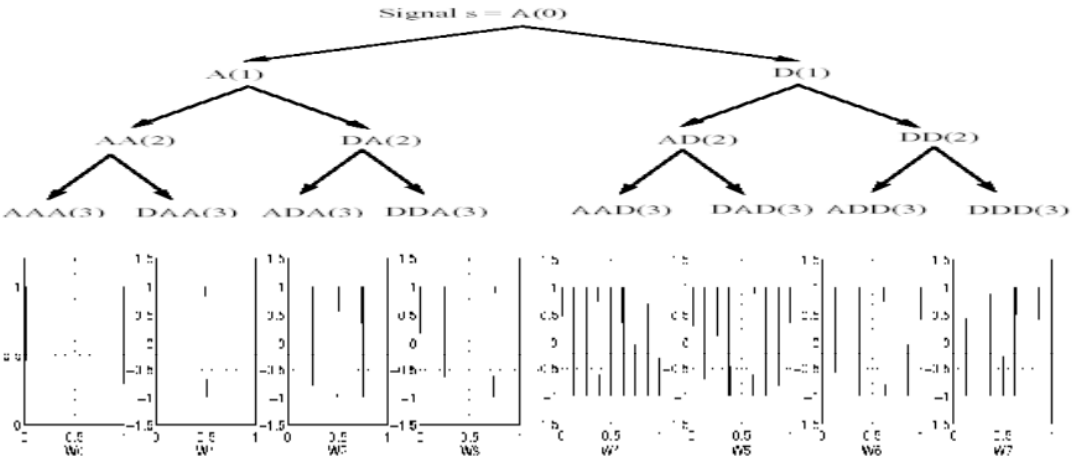
$$E(s) = -\sum_n s(n)^2 \log\left\lfloor s(n)^2 \right\rfloor$$

s konvenciou $0\log(0) = 0$.

Kritérium teda je:

Ak suma Shannonových entropií 2 subpásiem, ktoré vznikli rozdelením pôvodného subpásma, je menšia ako entropia pôvodného subpásma, je výhodné rozdelenie uskutočniť.

Ako vyzerajú bázové funkcie pri úplnom rozklade pre Haarovu WPT?



Počet prechodov nulou je:

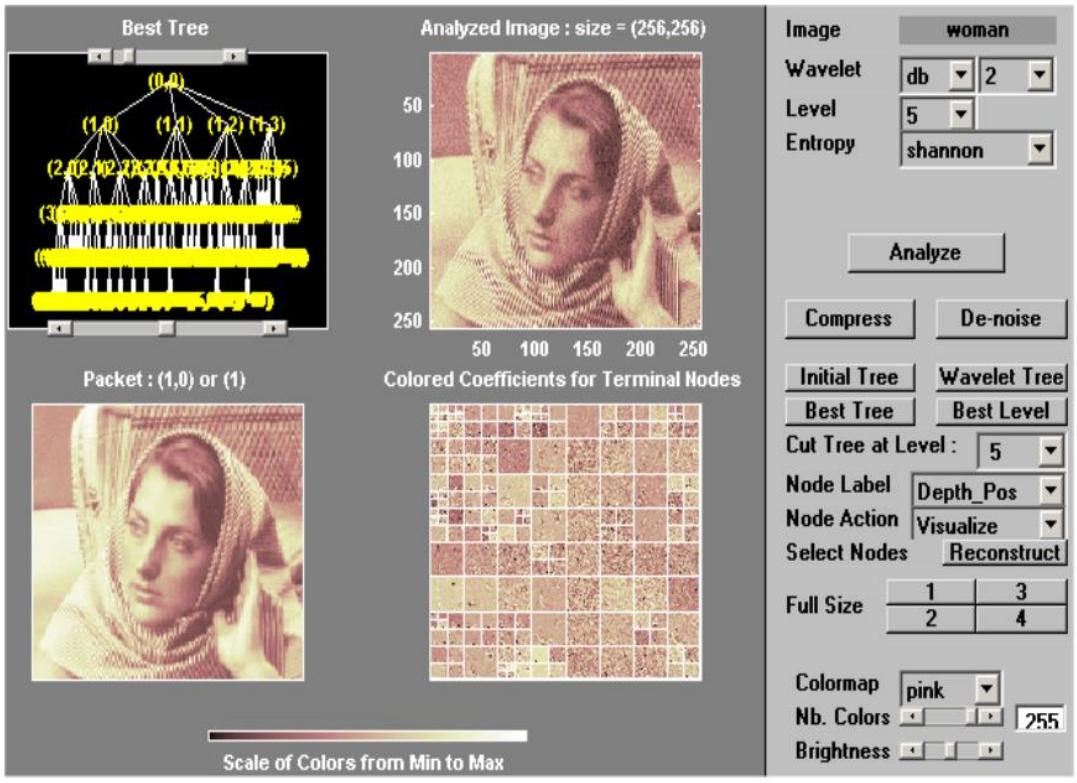
2 3 5 4 9 8 6 7

T.j. frekvenčné pásma odpovedajúce jednotlivým priestorom nemajú vzostupný charakter, sú poprehadzované, ich poradie je:

0 1 3 2 7 6 4 5

= analogicky ako pri *Walshovej transformácii v Palleyho poradí*, t.j. nie v sekvenčnom.

Chceme zobraziť frekvenčný obsah v prirodzenom poradí ? →*koeficienty preusporiadať*



Prednáška 4

DWT v maticovom tvare – otázku čakajte že napr. aký majú tvar tie matice. Viac nie iba aby sme tušili, že tam sú dve matice nad sebou čo máme na príkladoch vyskúšané

Maticový zápis pri periodickom rozšírení signálu

$$\mathbf{H}_m = \begin{pmatrix} h(0) & h(1) & \dots & \dots & \dots & h(-1) \\ \dots & h(-1) & h(0) & h(1) & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \dots & \dots & \dots & h(-1) & h(0) & h(1) \end{pmatrix} \quad \mathbf{G}_m = \begin{pmatrix} g(0) & g(1) & \dots & \dots & \dots & g(-1) \\ \dots & g(-1) & g(0) & g(1) & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \dots & \dots & \dots & g(-1) & g(0) & g(1) \end{pmatrix} \Big\} \frac{N_m}{2}$$

$$\mathbf{H}_m^T = N_m \begin{Bmatrix} h(0) & \vdots & \dots & \vdots \\ h(1) & h(-1) & \dots & \vdots \\ \vdots & h(0) & \dots & \vdots \\ \vdots & h(1) & \dots & h(-1) \\ \vdots & \vdots & \dots & h(0) \\ h(-1) & \vdots & \dots & h(1) \end{Bmatrix} \quad \mathbf{G}_m^T = N_m \begin{Bmatrix} g(0) & \vdots & \dots & \vdots \\ g(1) & g(-1) & \dots & \vdots \\ \vdots & g(0) & \dots & \vdots \\ \vdots & g(1) & \dots & g(-1) \\ \vdots & \vdots & \dots & g(0) \\ g(-1) & \vdots & \dots & g(1) \end{Bmatrix}$$

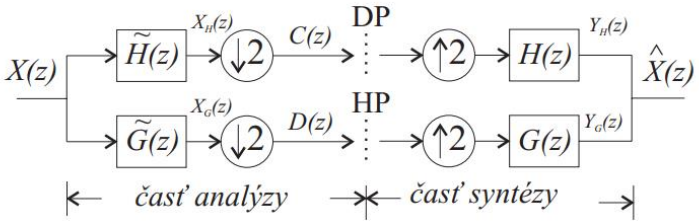
Analýza: $\mathbf{C}_{m+1} = \mathbf{H}_m \mathbf{C}_m \quad \mathbf{D}_{m+1} = \mathbf{G}_m \mathbf{C}_m$

$$\begin{pmatrix} \mathbf{C}_{m+1} \\ \mathbf{D}_{m+1} \end{pmatrix} = \begin{pmatrix} \mathbf{H}_m \\ \mathbf{G}_m \end{pmatrix} \mathbf{C}_m \quad \rightarrow \quad \mathbf{T}_a^{(m)} = \begin{pmatrix} \mathbf{H}_m \\ \mathbf{G}_m \end{pmatrix}$$

Syntéza: $\mathbf{C}_m = \begin{pmatrix} \mathbf{H}^T{}_m & \mathbf{G}^T{}_m \end{pmatrix} \begin{pmatrix} \mathbf{C}_{m+1} \\ \mathbf{D}_{m+1} \end{pmatrix} \quad \rightarrow \quad \mathbf{T}_s^{(m)} = \begin{pmatrix} \mathbf{H}^T{}_m & \mathbf{G}^T{}_m \end{pmatrix}$

Dvojpásmove banky filtrov – vysvetliť čo sa tam deje (konvolúcia, podvzorkovanie, nadvzorkovanie, analýza, syntéza)
A že pri konvolúcií sa deje to isté ako pri DWT len sa tam otáčajú filtre
Takže určite sa treba na to pozrieť

Dvojpásmové banky filtrov



Platí(V1): $c(n) = \sum_k \tilde{h}(2n-k)x(k) \qquad d(n) = \sum_k \tilde{g}(n-2k)x(k)$

$$\hat{x}(n) = \sum_k h(n-2k)c(k) + \sum_k g(n-2k)d(k)$$

DWT(V2): $c_{m+1}(n) = \sum_k \tilde{h}_{mr}(k-2n)c_m(k) \qquad d_{m+1}(n) = \sum_k \tilde{g}_{mr}(k-2n)c_m(k)$

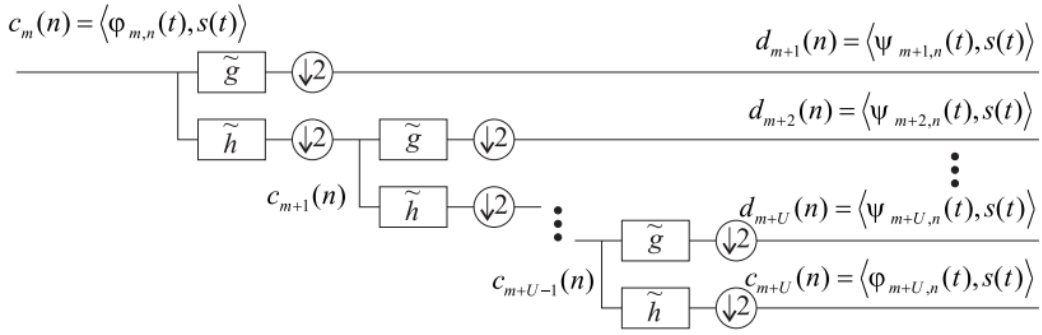
$$c_m(n) = \sum_k h_{mr}(n-2k)c_{m+1}(k) + \sum_k g_{mr}(n-2k)d_{m+1}(k)$$

T.J.AK PLATÍ $\tilde{h}(n) = \tilde{h}_{mr}(-n) \qquad h(n) = h_{mr}(n)$

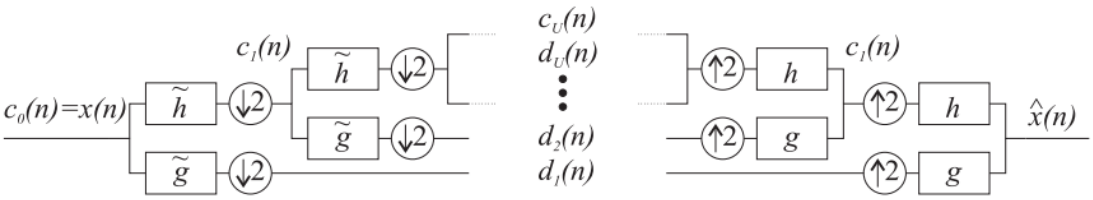
$\tilde{g}(n) = \tilde{g}_{mr}(-n) \qquad g(n) = g_{mr}(n)$ SÚ V1 a V2 TOTOŽNÉ !!!

Vysvetliť, že to vlastne opakujeme, tak ako sme iterovali tie matice tak takto iterujeme banku filtrov – tu je to graficky znázornené

Realizácia výpočtu WR pomocou bánk filtrov:



Realizácia výpočtu DWT a IDWT pomocou bánk filtrov:



Princíp kaskádeového algoritmu

Vzťah medzi h a g pri ortogonálnych waveletoch:

$$f(t)=\sum_m \sum_n d_{m,n}\tilde{\Psi}_{mn}(t)$$

Generovanie $\varphi(t)=\sum_n h_{mr}(n)\varphi(2t-n)$, z $\psi(t)=\sum_n g_{mr}(n)\varphi(2t-n)$ - kaskádové algoritmy

Ako vypočítať $\varphi(t)$ a $\psi(t)$ ak poznáme koeficienty pre zmenu mierky?

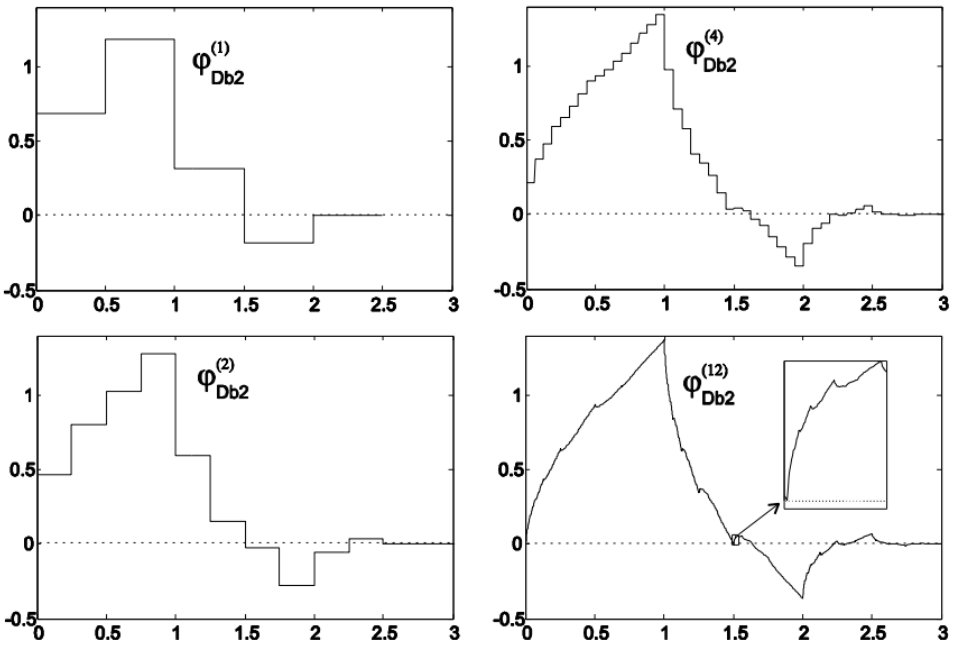
Vychádzajme z rovníc:

$$\varphi(t)=\sqrt{2}\sum_{n=-\infty}^{\infty} h_{mr}(n)\varphi(2t-n) \qquad \psi(t)=\sqrt{2}\sum_{n=-\infty}^{\infty} g_{mr}(n)\varphi(2t-n)$$

Tieto rovnice môžeme riešiť *iteračne*, pričom ak postup bude konvergovať k pevnému bodu, potom je pevný bod hľadaným riešením. Iterácie sú definované:

$$\varphi^{(k+1)}(t)=\sqrt{2}\sum_{n=-\infty}^{\infty} h_{mr}(n)\varphi^{(k)}(2t-n) \qquad \psi^{(k+1)}(t)=\sqrt{2}\sum_{n=-\infty}^{\infty} h_{mr}(n)\psi^{(k)}(2t-n)$$

Uvedený iteračný postup sa nazýva aj *kaskádový algoritmus*.



Generovanie funkcie mierky Db2 waveletu kaskádovým algoritmom v čase. Výpočet zobrazený po 1,2,4 a 12 iteráciách. Počiatočný signál bola "Box" funkcia. Vpravo dole je zo zväčšeniny zrejmý fraktálový charakter. Porovnajte s tvarmi bázových funkcií priestorov V

Prednáška 3

Vlastnosti SWT – treba vedieť všetky 3

Vlastnosti SWT

Linearita:

Vyplýva priamo z linearity skalárneho súčinu

Posun v čase:

$$g(t)=f(t-b_0) \Rightarrow SWT_g(a,b)=SWT_f(a,b-b_0)$$

Zmena mierky:

$$g(t)=\frac{1}{\sqrt{s}}f\left(\frac{t}{s}\right) \Rightarrow SWT_g(a,b)=SWT_f\left(\frac{a}{s},\frac{b}{s}\right)$$

Základné charakteristiky waveletov

1) *Existencia nosiča* na intervale vypovedá o tom, že funkcia (wavelet) má nenulové funkčné hodnoty len na danom intervale. Ak sú mimo neho funkčné hodnoty presne nulové, ovoríme, že na danom intervale $\langle a,b\rangle$ *má kompaktný nosič*. Ak sú „približne" nenulové, t.j. zanedbateľne malé hovoríme, že má *efektívny nosič*

2) *Počet nulových momentov*. *K-ty moment* $\psi(t)$ definujeme ako $m(k)=\int t^k\psi(t)dt$. Platí, že ak $\psi(t)$ je *K* krát *diferencovateľná* a pre $t\to\pm\infty$ klesá dostatočne rýchlo, potom prvých *K-1* momentov bude nulových. Potom ak $f(t)$ je na nejakom intervale polynómom max. *K-1* stupňa, pre wavelety $\psi_{a,b}(t)$ podporované v tomto intervale budú príslušné waveletové koeficienty $SWT_f(a,b)$ nulové.

3) *Regularita* (Daubechies 1988) poskytuje *mieru hladkosti funkcie* $f(t)$. Je to také maximálne číslo *r* pre ktoré platí $|F(\omega)|\le c/\left(1+|\omega|^{r+1}\right)$, $\omega\in R$. Potom $f(t)$ je *r-1* krát spojite diferencovateľná, *r-tá* derivácia môže byť nespojitá.

Vzorec na výpočet waveletovych radov

$$f(t)=\sum_m\sum_n d_{m,n}\tilde{\psi}_{m,n}(t) \qquad d_{m,n}=\left\langle f(t),\psi_{m,n}(t)\right\rangle \qquad m,n\in Z$$

Algoritmus výpočtu Waveletových radov – treba vedieť vysvetliť

Algoritmus výpočtu Waveletových radov a disktrétnej waveletovej transformácie

Nech funkcia $s(t) \in L(R)$. Potom súradnice jej *priemetu* do priestoru V_m môžeme vyjadriť ako:

$$c_m(n) = \langle s(t), \varphi_{m,n}(t) \rangle$$

Získavame diskrétnu množinu ***koeficientov mierky*** $c_m(n)$.
Aká informácia sa do nich uložila? → spravme spätnú rekonštrukciu

$$\hat{s}_{V_m}(t) = \sum_{n=-\infty}^{\infty} c_m(n) \varphi_{m,n}(t)$$

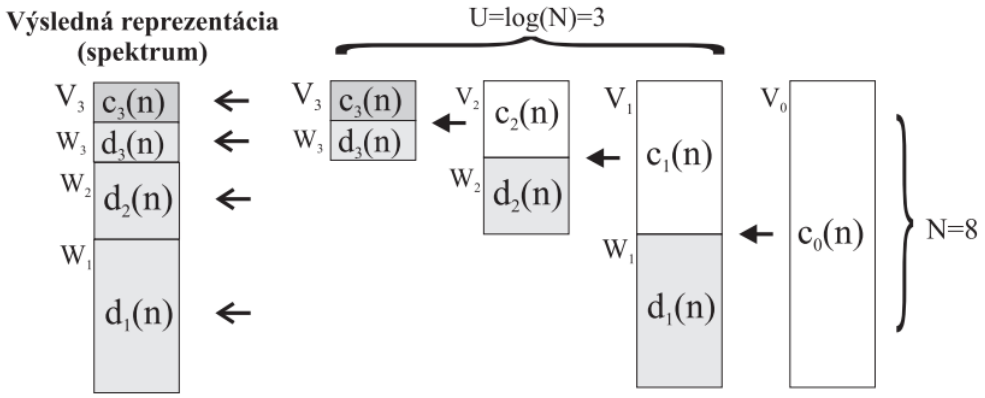
Získavame $\hat{s}_{V_m}(t) \in V_m$ - *aproximáciu* (ap_m) funkcie $s(t)$ vo V_m.

Analogicky priemetom $s(t)$ do W_m získame ***waveletové koeficienty*** $d_m(n)$:

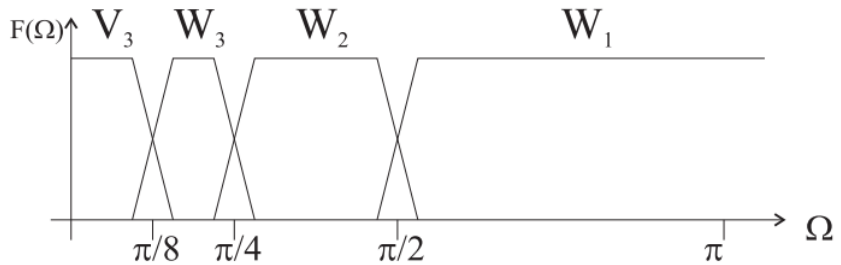
$$d_m(n) = \langle s(t), \psi_{m,n}(t) \rangle$$

Aká informácia sa do nich uložila? → spravme spätnú rekonštrukciu

Vedieť vysvetliť, aké je rozdelenie subpásiem a prečo to tak je (kľúčova vec)



Výpočet koeficientov pri DWT a veľkosti vstupného vektora 8.



Označme diskrétny vstup pre DWT ako f(n). Na príklade jeho DTFT f(Ω) je znázornené, ktoré časti spektra budú vyjadrené v ktorých podpriestoroch.

Prednáška 2

Reprezentácia signálu v čase – vraj sme s tým rátali tak by sme to mali vedieť. Máme si pozrieť aj tie vzorce, ale len že ako vyzerá prvy a druhý moment na skúške ich cele netreba napísať skôr len aby sme vedeli, že to existuje

Reprezentácia signálu v čase a frekvencii

Pri analýze a reprezentácii signálov je častokrát výhodné použiť transformáciu, ktorá reprezentuje signál súčasne v čase aj frekvencii.

- Riešenie vo forme oknovej *STFT (Short Time Fourier Transform),* Gábor (1946), posúva okno fixnej veľkosti pozdĺž signálu a extrahuje frekvenčný obsah v danom intervale

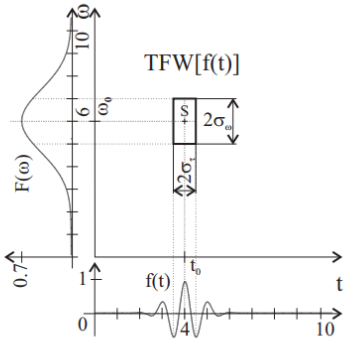
$$F(\omega,\tau)=\int_{-\infty}^{\infty} f(t)g(t-\tau)e^{-j\omega t}dt=\left\langle f(t)g(t-\tau),e^{j\omega t}\right\rangle$$

,kde $g(t)$ je *oknová funkcia* a $f(t)$ vstupná funkcia. Ak oknová funkcia je gaussovská funkcia, tak STFT sa volá *Gáborova transformácia*.

Pre STFT platí:

- Bázové funkcie sú generované *moduláciou* a *posunom* oknovej (prototypovej) funkcie $g(t)$.
- STFT má pre danú oknovú funkciu *pevné rozlíšenie vo frekvencii*.
- Nadbytočnosť STFT môžeme odstrániť vzorkovaním

Každý signál môže byť znázornený v *časovo-frekvenčnej rovine(TF rovina)*, ktorá charakterizuje rozdelenie napr. energie signálu v čase a frekvencii:



$$x(t)=e^{-(t-4)^2}\cos(6(t-4))$$

Rozlíšenie v čase a frekvencii pre danú funkciu $x(t)$ a jej fourierovu transformáciu $X(\omega)$ je dané *časovo – frekvenčným oknom (TF okno)*. Jeho stred je v bode $S=(t_0,\omega_0)$, a veľkosti strán sú $2\sigma_t,2\sigma_\omega$. (ω_0 sa nazýva stredná frekvencia signálu). Zobrazuje sa v tzv. *TF rovine*.

Platí:

$$t_0=\|x(t)\|^{-2}\int_{-\infty}^{\infty} t\,|x(t)|^2\,dt \qquad \omega_0=\|X(\omega)\|^{-2}\int_{0}^{\infty}\omega\,|X(\omega)|^2\,d\omega \quad \text{(1.moment)}$$

$$\sigma_t^2=\|x(t)\|^{-2}\int_{-\infty}^{\infty}(t-t_0)^2|x(t)|^2\,dt \qquad \sigma_\omega^2=\|X(\omega)\|^{-2}\int_{0}^{\infty}(\omega-\omega_0)^2\,|X(\omega)|^2\,d\omega \quad \text{(2.moment)}$$

Princíp neurčitosti - vedieť

Princíp neurčitosti:

Pre $x(t)$, ktoré spĺňa $t \cdot x(t) \in L^2(R)$, platí:

$$\sigma_t^2 \cdot \sigma_\varpi^2 \geq 1/4$$

Čo je to spojitá waveletová transofrmácia

Spojitá waveletová transformácia (SWT) funkcie $f(t) \in L^2(R)$ je definovaná ako zobrazenie $L^2(R) \to L^2(R^2)$ vzťahom

$$SWT_f(a,b) = \int_{-\infty}^{\infty} f(t)\psi_{a,b}^*(t)dt = \langle f(t), \psi_{a,b}(t) \rangle \qquad a \in R^+, b \in R$$

Funkcie $\psi_{a,b}(t)$ sú definované zo *základného waveletu* $\psi(t)$ pomocou parametrov *zmeny mierky* a *posunu* a, b nasledovne:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right), \qquad \psi(t) \in L^2(R)$$

SWT v závislosti od parametra a poskytuje pružné časovo - frekvenčné rozlíšenie. Ak *časovo - frekvenčné okno* $\psi(t)$ má rozmery σ_t, σ_ϖ a stred v bode $S = (t_0, \omega_0)$, potom

$$\sigma_{ab_t} = a\sigma_t \qquad \sigma_{ab_\omega} = \sigma_\omega / a \qquad S_{ab} = (at_0 + b, \omega_0 / a)$$

T.j. obsah okna ostáva konštantne $4\sigma_t\sigma_\omega$.

Základna podmienka pre Wavelet – Nesmie mať jednosmernú zložku (vyjadrené matematicky)

Wavelet je potom ***prípustný*** a platí

$$\Psi(0) = \int_{-\infty}^{\infty} \psi(t)dt = 0$$

Jednotková energia waveletu

Pozn.1:Základné wavelety majú zvyčajne jednotkovú energiu, t.j. $\|\psi(t)\| = 1$.

Pozn.2: Normalizácia $1/\sqrt{a}$ pri výpočte $\psi_{a,b}(t)$ z $\psi(t)$ zabezpečuje $\|\psi_{a,b}(t)\| = \|\psi(t)\|$.

Prednáška 1

Hilbertove priestory – definície priestorov

Separabilné Hilbertove priestory

Komplexné / reálne priestory

Komplexný priestor C^n je množina všetkých n-tíc $x=(x_1,x_2,...,x_n)$ s **konečnými hodnotami** x_i na množine C. *Skalárny súčin* je definovaný ako:

$$\langle x,y\rangle=\sum_{i=1}^{n}x_i y_i^* \qquad x,y\in C^n$$

Analogická definícia platí aj pre R^n. Kvôli jednoznačnosti budeme používať aj klasickú notáciu $\bar{x}=(x_1\ x_2\ \dots\ x_n)^T$.

Priestor $l^2(Z)$

Vektormi x v priestore $l^2(Z)$ sú sekvencie $x(n)\in C$, $n\in Z$, s konečnou energiou $\|x\|<\infty$. Zvyčajne reprezentujú signály distkrétne v čase. *Skalárny súčin* je definovaný ako:

$$\langle x,y\rangle=\sum_{n=-\infty}^{\infty}x(n)y(n)^* \qquad x,y\in l_2(Z)$$

Priestor $L^2(R)$

Vektormi x v priestore $L^2(R)$ sú funkcie $x(t)\in C$, $t\in R$, ktoré sú po štvorcoch integrovateľné, t.j. $\|x\|<\infty$. *Skalárny súčin* je definovaný ako:

$$\langle x,y\rangle=\int_{t\in R}x(t)y(t)^*\,dt \qquad x,y\in L_2(R)$$

Pozn.: Analogicky pre funkcie n premenných môžeme definovať priestory $L_2(R^n)$

Ortonormálne bázy – to sme rátali máme vedieť

Ortonormálne bázy

Množina $B = \{b_i\}$ je *ortonormálny systém* v priestore E.ak

$$\forall b_i, b_j \in B;\ \langle b_i, b_j \rangle = \delta(i-j)$$

$B = \{b_i\}$ je bázou priestoru E, ak všetky $y \in E$ môžeme vyjadriť

$$y = \sum_k \alpha_k b_k$$

kde α_k sú *spektrálne koeficienty*

$$\alpha_k = \langle b_k, y \rangle .$$

Pre takýto systém platí *Parsevalova rovnosť*

$$\|y\|^2 = \sum_i |\langle b_i, y \rangle|^2 \qquad \forall y \in E$$

Analogické tvrdenia platia aj pre *ortogonálne systémy*, vo vzťahoch je iba pridaná normalizačná konštanta.