

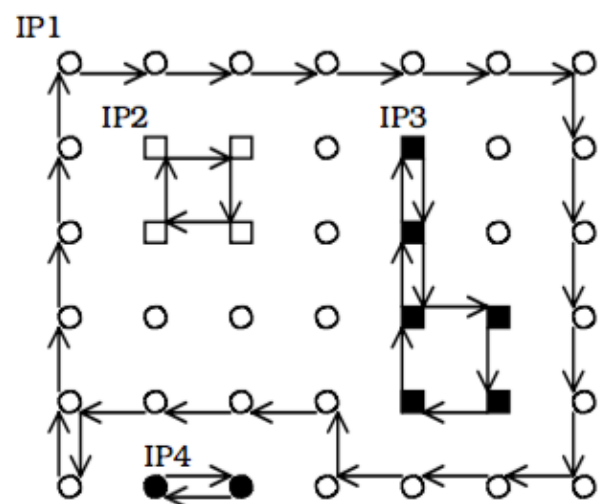
Viacúrovňové kódovanie

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 jasú (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 najúčinnnejší pri malom počte kvantizačných úrovní a veľkých jednoúrovňových objektoch.

Viacúrovňové kódovanie – tabuľky

poradie obrysu	kódové slovo	riadky a stĺpce	kódové slovo
1	00	1	000
2	01	2	001
3	10	:	
4	11	8	111
úroveň	kódové slovo	smer popisu	kódové slovo
○	00	↑	00
□	01	→	01
■	10	↓	10
●	11	←	11

Príklad kódovania:

00 –poradie obrysu IP1 (na spodku), 00-úroveň jasů „○“, 000-x, 000-y, 01-posun1, 01-posun2, 01, 01, ...,00, 00 (dorazili sme do počiatočných súradníc, tu je koniec) , 01 – poradie obrysu IP2 (nad IP1), ...

Kódovanie tvarov (videoobjektov) v MPEG4 – reťazový kód

Pri tejto metóde sú hranice objektu reprezentované uzavretou linkou. Kódovanie spočíva v zaznamenávaní smeru pohybu tejto linky. Kódovanie je ukončené, keď je dosiahnutý východzí bod.

Zakódované dáta určujú štartovací bod s prvým smerovým kódom a nasledujúce diferencne zakódované smery. Ak VOP obsahuje viacero uzavretých liniek, tak reťazové kódy nasledujú za informáciou o počte oblastí.

Keďže reťazový kód je cyklický, smer môžeme diferencne kódovať do hodnôt od -3 do 4:

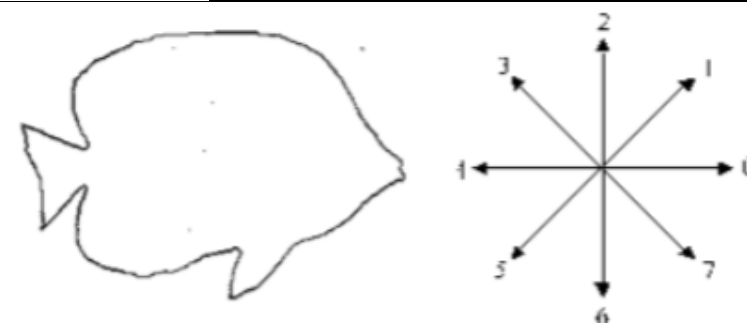
$$\begin{aligned} &cn - cn-1 + 8, && \text{ak } cn - cn-1 > -3 \\ d = &cn - cn-1 - 8, && \text{ak } cn - cn-1 < -4 \\ &cn - cn-1, && \text{inak} \end{aligned} \quad (4.1)$$

kde d je diferencný reťazový kód, cn súčasný a $cn-1$ predchádzajúci smer. Na kódovanie d sa používa Huffmanov kód.

Huffmanov kód pre diferencný reťazový kód

d	kód
0	1
1	00
-1	011
2	0100
-2	01011
3	010100
-3	0101011
4	0101010

Poznámka: Pozor, v texte vyššie sú operátory
< > naopak!



Obr. 6.7 Kódovaný objekt a znázornenie ôsmich kódovacích smerov [64]

Na strane prijímača je cn dekódovaná podľa vzorca

$$cn = (cn-1 + d + 8) \bmod 8$$

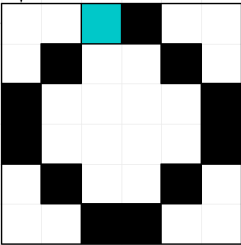
Príklad:

Objekt

(x,y)=(0,0)

↓

→



Kódovanie hranice , začíname v bode (x, y) = (2, 0)

Začíname doprava (výsledok má 26 bitov)

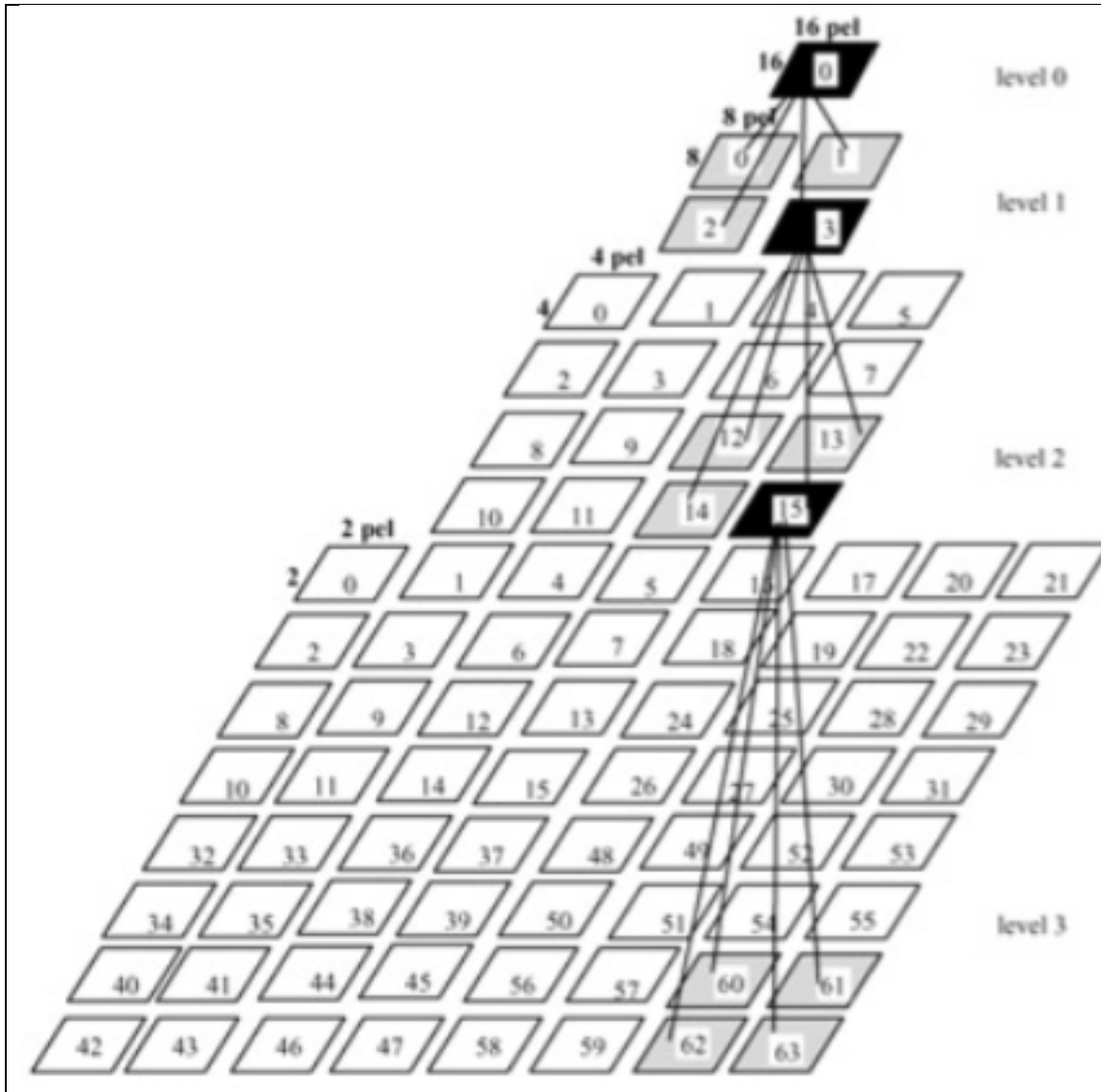
N	0	1	2	3	4	5	6	7	8	9	10	11
c_n	0	7	7	6	5	5	4	3	3	2	1	1
c_n-c_n-1	0	7	0	-1	-1	0	-1	-1	0	-1	-1	0
d_n [-3,4]	0	-1	0	-1	-1	0	-1	-1	0	-1	-1	0
kód	1	011	1	011	011	1	011	011	1	011	011	1

Začíname doľava (výsledok má 25 bitov)

N	0	1	2	3	4	5	6	7	8	9	10	11
c_n	5	5	6	7	7	0	1	1	2	3	3	4
c_n-c_n-1	5	0	1	1	0	-7	1	0	1	1	0	1
d_n [-3,4]	-3	0	1	1	0	1	1	0	1	1	0	1
kód	0101011	1	00	00	1	00	00	1	00	00	1	00

Kódovanie tvarov (videoobjektov) v MPEG4 – quad tree kódovanie

Vstup – tzv. BAB (Binárny Alfa Blok) bloky – veľkosti 16x16 (binárne hodnoty: 0=čierna, 255=biela)



BAB blok vyjadruje hodnotami:

- kde objekt je
- kde objekt nie je

Každý BAB blok je vyjadrený hierarchicky:

- V úrovni 3 (L3) je rozdelený na 64 blokov, rozmeru 2x2
- V úrovni 2 (L2) je rozdelený na 16 blokov zo 4x4 pixelmi
- V úrovni 1 (L1) je rozdelený na 4 bloky zo 8x8 pixelmi
- V úrovni 0 (L0) je rozdelený na 16 blokov zo 16x16

Každý blok na každej úrovni je kódovaný pomocou čísla označovaného ako „index“.

Quad tree úroveň 3

b[0]	b[1]
b[2]	b[3]

Vypočítanie indexu pre každý zo 64 blokov:

$$\text{IndexL3} = 27b[0] + 9b[1] + 3b[2] + 1b[3]$$

pričom

- $b[i]=2$ ak pixel=255, $b[i]=0$ ak pixel=0
- dostávame takto 16 rôznych hodnôt indexu (min=0, max=80)

	horný_ index[0]	horný_ index[0]
ľavý_ index[0]	index[0]	index[1]
ľavý_ index[0]	index[2]	index[3]

Poprehadzovanie indexov (zvýšenie medziblokovej korelácie) v rámci štvorice indexov na základe situácie nad a vľavo od tejto štvorice:

Ak $\text{horný_index}[0] < \text{horný_index}[1]$, vymeň $\text{index}[0]$ s $\text{index}[1]$ a $\text{index}[2]$ s $\text{index}[3]$ okrem sub-blokov 0,1,4,5,16,17,20 a 21

Ak $\text{ľavý_index}[0] < \text{ľavý_index}[1]$, vymeň $\text{index}[0]$ s $\text{index}[2]$ a $\text{index}[1]$ s $\text{index}[3]$ okrem sub-blokov 0,2,8,10,32,34,40 a 42

Ak $\text{horný_index}[0] + \text{horný_index}[1] < \text{ľavý_index}[0] + \text{ľavý_index}[1]$, vymeň $\text{index}[1]$ s $\text{index}[2]$ 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 $\text{Index_L2} = 27f(\text{indexL3}[0]) + 9f(\text{indexL3}[1]) + 3f(\text{indexL3}[2]) + 1f(\text{indexL3}[3])$ 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 niektorých výmenách.

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

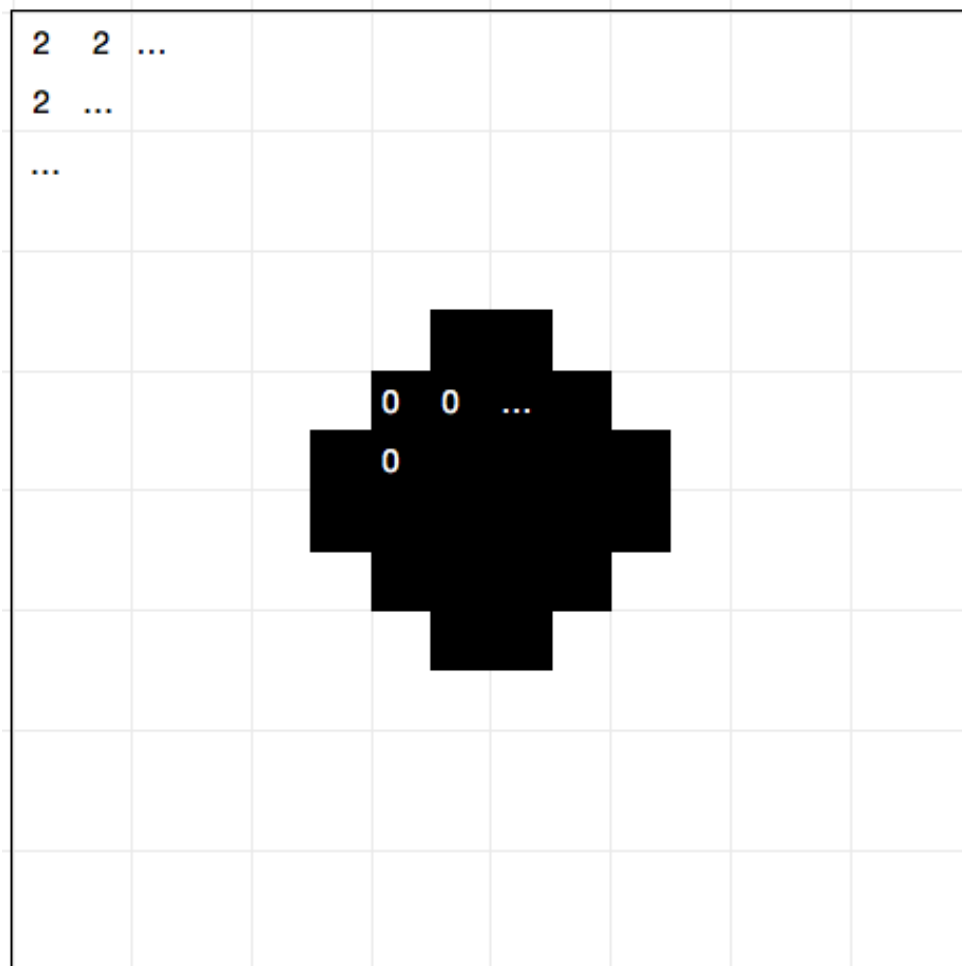
Analogický obrázok ako pri L3.	Vypočítanie 1 indexu zo 4 indexov úrovne 2 ako na 1 úrovni
--------------------------------	---

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)

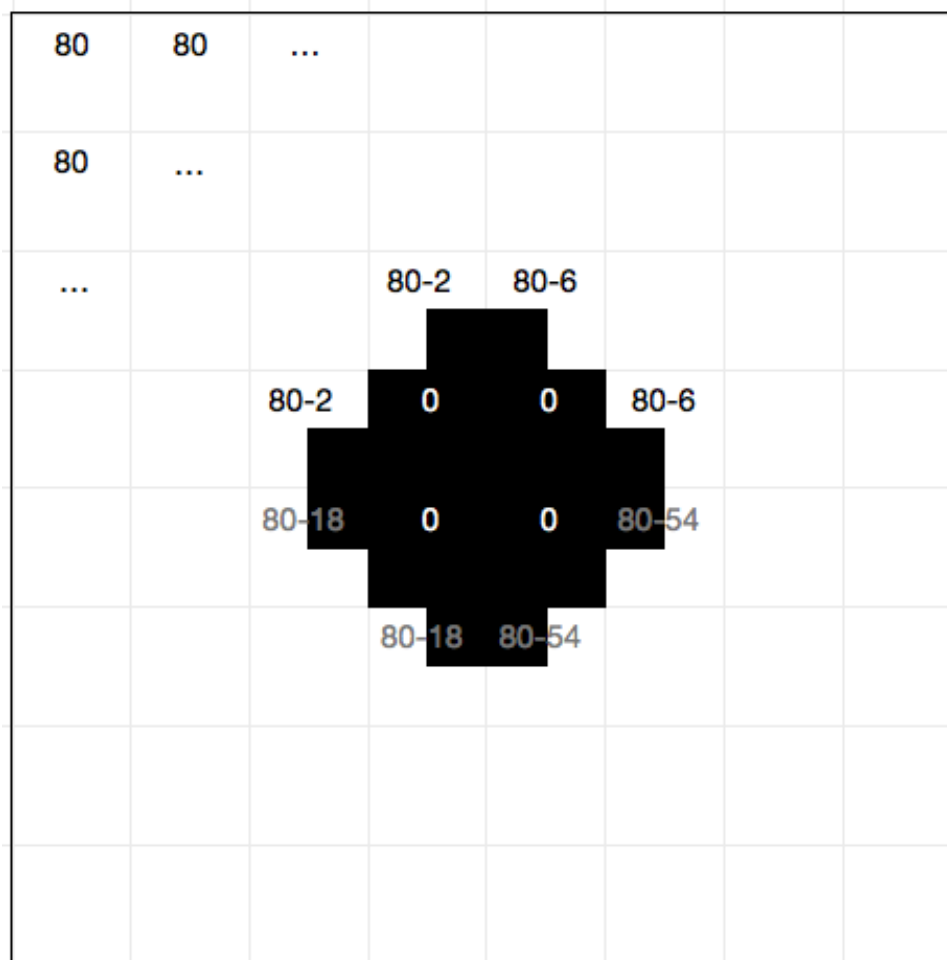
Príklad

BAB blok 16x16 s naznačenými b hodnotami
v leveli 3



Odpovedajúce indexy L3

- $27 \times 2 + 9 \times 2 + 3 \times 2 + 2 = 80$
- týchto 64 indexov sa kóduje pomocou HK



Indexy – nepovymienené v L3

80	80	...					
80	...						
...			80-2	80-6			
			80-2	0	0	80-6	
			80-18	0	0	80-54	
				80-18	80-54		

Indexy – povymienené v L3

- v rámci belasého kvadrantu došlo k indexov vo vertikálnom aj horizontálnom smere (prvé 2 podmienky)

80	80	...					
80	...						
...			80-2	80-6			
			80-2	0	0	80-6	
			80-18	0	80	80-54	
				80-18	80-54	0	

Následne sa pokračuje počítaním indexov pre L2 na základe povymieňaných L3 indexov

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 efektívnosť
 - aj makrobloky môžu byť až 64x64



- 33+2 smerov pre kompenzáciu pohybov (namiesto 9)
- adaptívna predikcia vektorov pohybu
- používa CABAC ako hlavný entropický kódér (a používa aj Golomb-Rice a Exp. Golombove kódy)
- zvýšené možnosti paralelizmu (tiles, slices)
- ...

HEVC – efektivita kódovania oproti H.264/MPEG2 AVC a VP9

- zdvojnásobuje kompresný pomer?
- Subjektívne porovnanie [[Tak-Mrak-2014](#)]:

	H.264/MPEG4 AVC			
	480p	720p	1080p	4K UHD
Redukcia bitovej náročnosti pre HEVC	52%	56%	62%	64%

- Objektívne (založené na PSNR) porovnanie
 - O cca. 35% [[Ohm, Sullivan-2012](#)]

	H.264/MPEG4 AVC	MPEG4 ASP	H.263	MPEG2
Redukcia bitovej náročnosti pre HEVC	35.4%	63.7%	65.1%	70.8%

- Porovnanie s VP9 (formát vyvinutý googlom)
 - 49.5% redukcia

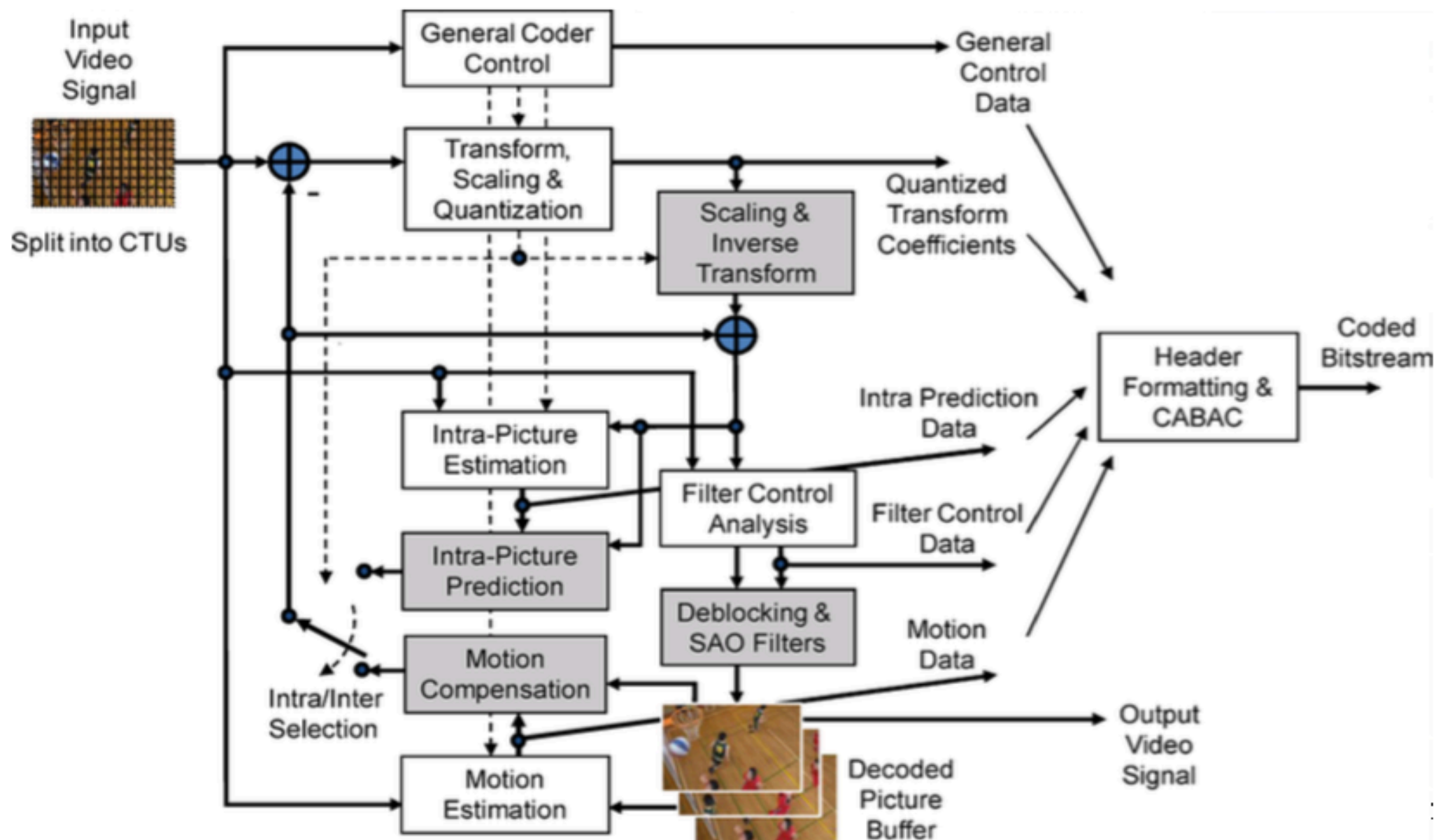
H265 obsahuje aj profil na kódovanie statického obrazu

- Profil: Main Still Picture (MSP)
- Štúdie [[Hantart2013](#)], [[Ugur2013](#)] ukazujú, že je efektívnejší ako JPEG 2000
 - Redukcia bitovej náročnosti 20% (PSNR), 30% (MOS)

Poznámka k patentovaniu:

- každý kto chce používať HEVC musí platiť tzv. „royalties“ držiteľom patentov, t.j. skupinám MPEG LA, HEVC Advance
- UHD Konkurencia (sú patentované, ale pre voľné použitie):
 - VP9 (Google) – „royalty free“, BSD licencia, WebM a Matroska kontajner
 - Daala (Xiph.org) – na voľné použitie, BSD licencia, Ogg kontajner

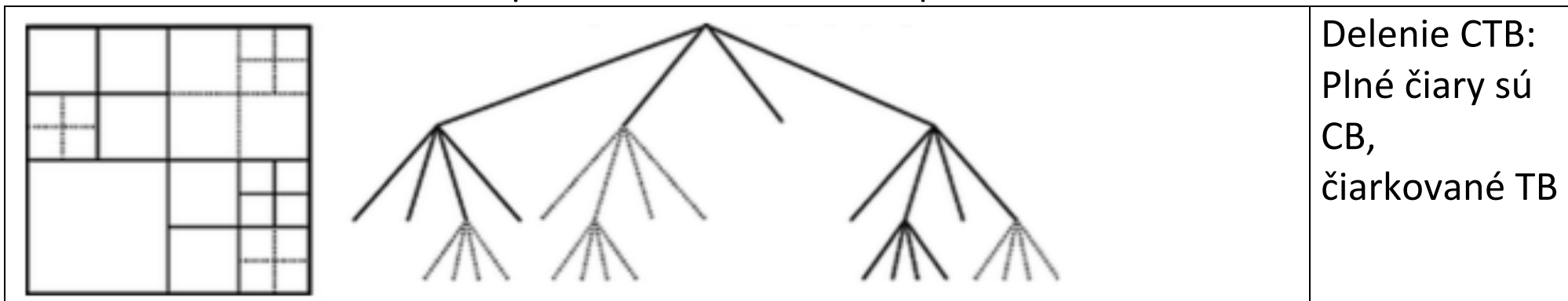
Bloková schéma H.265 enkódera



Pozn.: Sivé časti modelujú dekóder

Delenie na bloky v HEVC ...

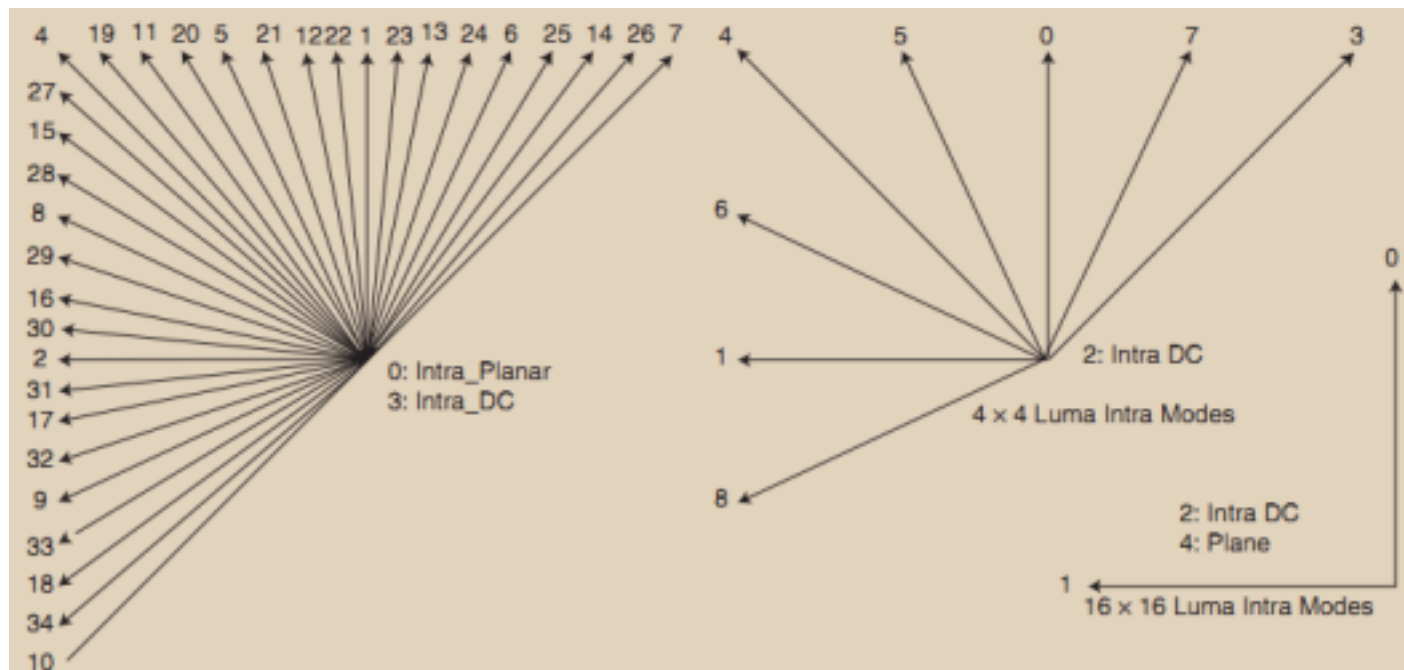
- obraz sa rozdelí na CTU
 - každá CTU sa logicky skladá z CTB (coding tree block) blokov pre jasovú a jednotlivé chromatické zložky (YCbCr), každá sa vzorkuje ináč (typicky 4:2:0 – polovičný počet bodov v každom smere pre obe chrominančné zložky)
 - CTB majú rozmery 16x16, 32x32, 64x64
 - CTB sa delia na Coding blocks (CB) logicky formujú CU (coding units)
 - CB sa delia na PB (predistion blocks) logicky formujú PU (prediction units) (4x4 až 64x64)
 - CB pri kódovaní rezidua sa **rekurzívne** delia na quadtree zložené z TB (transfrom blocks), ktoré logicky formujú TU (Transform units)
 - TB majú rozmery 4x4 až 32x32 (DCT, alebo DST transformácia + Kvantizácia)
 - DST sa používa ako celočíselná pre kódovanie luminancie



CABAC v H.265

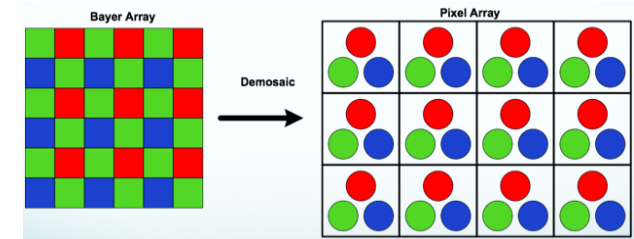
- menej kontextov ako v H.264
- menej závislostí, avšak stále efektívnejší ako v H. 264

Definícia smerov na kompenzáciu pohybu (vľavo H.265, vpravo H. 264)



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 ⁵
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