DEPARTMENT OF
NETWORKED SYSTEMS
AND SERVICES

Veszteséges forráskódolás

VIHIBB01 – Coding and IT Security, 2020

István Vajda

CrySyS Lab, BME
vajda@crysys.hu

www.crysys.hu

MŰEGYETEM 1782

Tartalom

- A forráskódolás célja és fajtái
- A veszteséges tömörítés alkalmazási területei
- Numerikus példák
- Torzítás mértékek
- JPEG
    - Kódolási séma
    - DCT
    - Kvantálás
    - Zig-zag letapogatás
    - DPCM, RLC
    - Entrópia kódolás
- MPEG

A forráskódolás célja

Cél az információ hatékony továbbítása vagy tárolása:

- több információ átvitele adott csatornán, több információ tárolása adott tárhelyen
- sávszélesség, tárkapacitás igény csökkentés

A forráskódolás két fajtája:

Veszteségmentes forráskódolás: Az eredeti adat tökéletesen visszaállítható a tömörítettből.

Veszteséges forráskódolás: A visszaállított adat csak az eredeti egy approximációja. Tipikusan lényegesen kedvezőbb tömörítési arányt tesz lehetővé mint a veszteségmentes eset. Elsősorban audió, kép, videó tömörítésére használják.

(tömörítési arány = tömörített folyam (bit/sec) / eredeti folyam (bit/sec))

Alkalmazási területek

Alkalmazások, amelyek nem léteznének forráskódolás nélkül:

- Digitális televízió (DVB-T)
- Internet video streaming (YouTube)

Alkalmazások, amelyek gazdaságossá váltak forráskódolás alkalmazásával:

- Distribution of digital images
- High definition television (HDTV) over IPTV

Számtalan alkalmazás használ forráskódolást:

- Software distributed in compressed form
- Audio data compression (MP3, AAC)
- Mobile audio players (IPod,...) and mobile phones
- Audio download (ITunes) and streaming services (Internet radio)
- Digital images are typically compressed (JPEG)
- Pictures on web sites are compressed
- Digital video data compression (MPEG-2, H.264/AVC)
- Output of video cameras, optical discs (DVD)
- Video streaming (Youtube, Internet TV)

- **Az Interneten a továbbított adatbitek kb. 70% -a forráskódolt video.**

Numerikus példák(*)

- File tömörítés (text file, office dokumentum, program kód, ...)
- Tipikus példa: 80 MByte tömörítése 20 Mbyte-ra (**25%-ra**)
- Audio tömörítés
- Sztereo: mintavételi frekvencia: 44,1 kHz
- 16 bit/minta
- =) Nyers adat sebesség: 44,1x16x2 = 1,41 Mbit/s
- =) Tipikus adatsebesség tömörítéssel: 64 kbit/s (**4.5%-ra**)
- Kép tömörítés
- Kép méret: 3000x2000 minta (6 MegaPixel)
- 3 színkomponens (vörös, zöld, kék) és 1 byte (8 bit) / minta
- =) Nyers file méret: 3000x2000x3 = 18 MByte
- =) Tipikus tömörített méret: 1 MByte (**5.6%-ra**)
- Video tömörítés
- Képméret 1920x1080 pixel és keret sebesség 50 Hz
- 8 bit/minta
- 3 színkomponens (vörös, zöld, kék)
- =) Nyers adat sebesség: 1920x1080x8x50x3 = 2,49 Gbit/s
- =) Tipikus adatsebesség tömörítéssel: 12 Mbit/s (**0.5%-ra**)

(*) heiko.schwarz@hhi.fraunhofer.de

Torzítási mértékek

- Torzítási mérték definiálása szükséges veszteséges tömörítés esetén

- Gyakran az a torzítás, amelyet az ember a kódolt tartalomban észlel, nagyon nehezen mérhető mennyiség, mivel az ***emberi érzékelés*** jellemzői összetettek.

- Beszéd/audio vonatkozásában az ***emberi érzékelés modellek*** jóval pontosabbak mint a kép/videó vonatkozásában
- *Beszéd/audio:*
  - ***hallgatási tesztek***et használnak a kódolási eredmények szubjektív minőségének meghatározására
- *Kép/videó:*
  - ***megtekintési tesztek***et használnak a kódolási eredmények szubjektív minőségének meghatározására

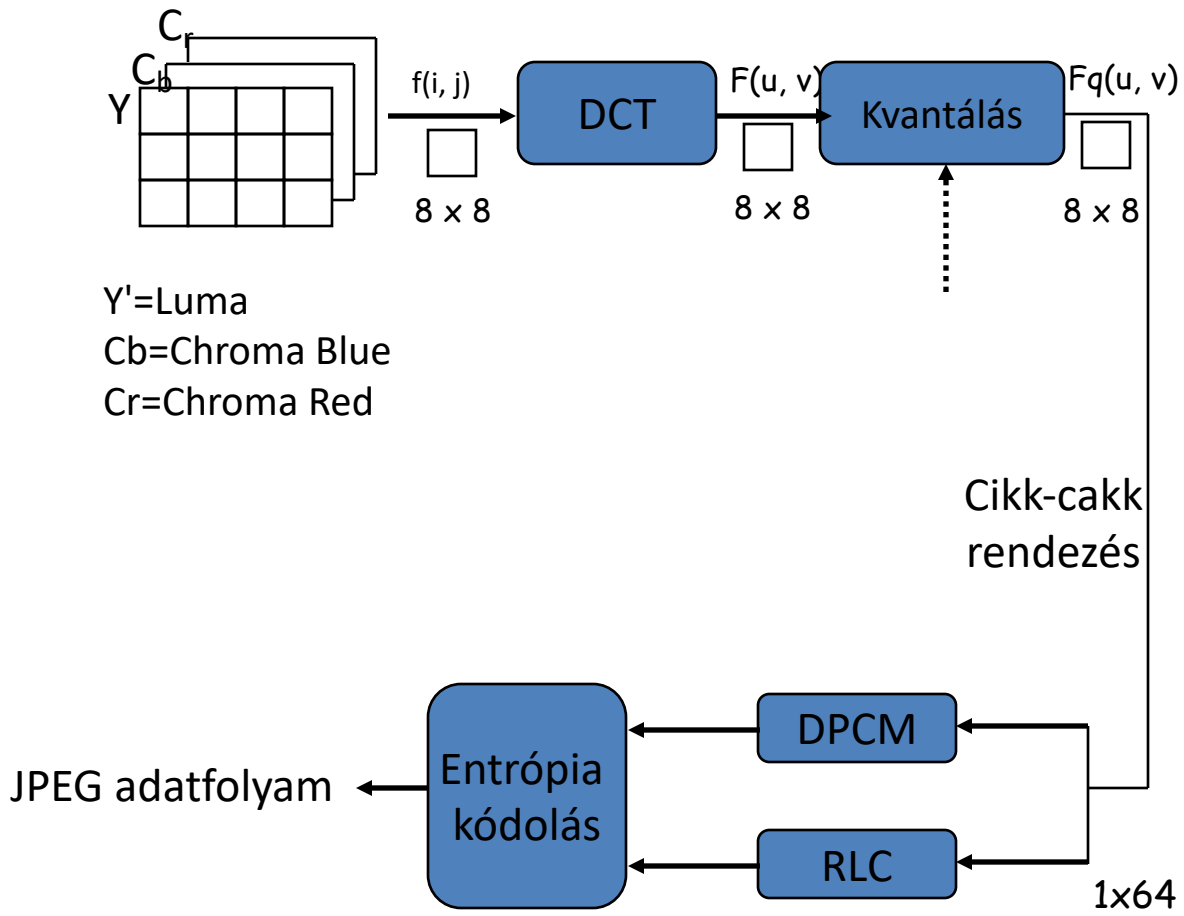
Képtömörítés: JPEG

Miért a JPEG?

- A tömörítési arány a veszteségmentes módszerek esetén (pl. Huffman, LZW) nem elegendő kép és videó tömörítéshez.

- JPEG ***transzformációs kódolást*** alkalmaz, amely lényegében a következő empirikus észrevételen alapul:
  - A képtartalom nagy része relatíve lassan változik a kép síkján, pl. kis képterületet tekintve az intenzitás értékek alig változnak.
  - A képsíkon számított kisfrekvenciás térkomponensek (lower spatial frequency components) több információt tartalmaznak mint a tipikusan lényegtelen részletekhez, zajhoz tartozó nagyfrekvenciás komponensek.
  - A tapasztalatok azt mutatják hogy az emberi érzékelés kevésbé érzékeny a nagyfrekvenciás mint a kisfrekvenciás komponensek eliminálására.

JPEG: kódolási séma



A kódolás lépései:

1. **Diszkrét Koszinusz Transzformáció (DCT)** 8x8 méretű pixel mátrixokra $f(x,y) \rightarrow_T F(u,v)$
2. **Kvantálás (Quantization)** kvantálási táblázat felhasználásával
3. **Cikk-cakk rendezés a** redundancia kihasználására
4. **Differenciális Pulzus Kód Moduláció** (**DPCM**) a DC komponensre and Futamhossz-kódolás (**RLC**) of the AC komponenre
5. Entrópia kódolás (**Huffman**) a végső outputhoz

A képet **képsík**okra bontjuk: Y (fényesség), C_b (kék), C_r (vörös)
1 **képpont**ot (pixel) képsíkonként 8 bittel ábrázolunk → 1 képpontot 3x8=24 bittel ábrázolunk
A képsíkokat 8x8 képpontot tartalmazó **blokk**okra bontjuk

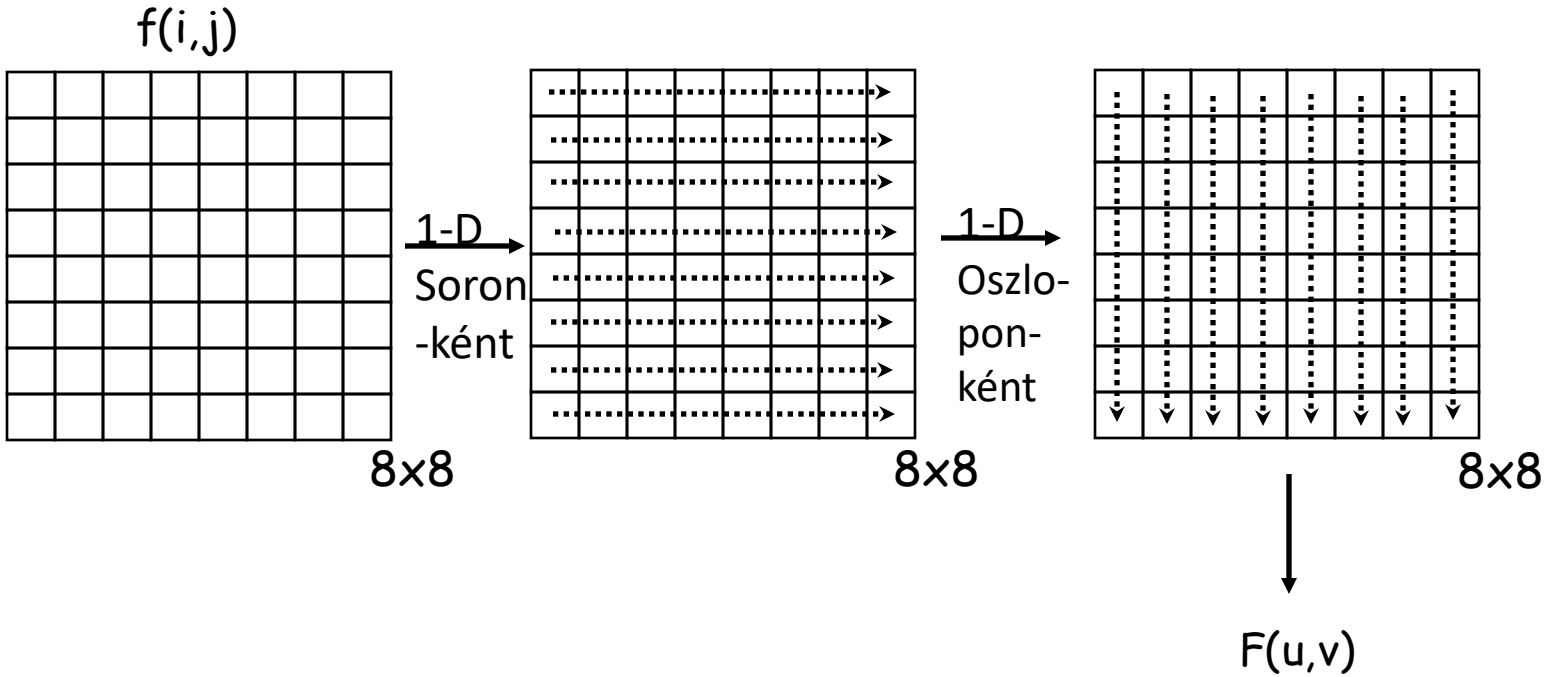
JPEG: Diszkrét Koszinusz Transformáció

DCT transzformáció a kép átalakítja a pixelek egy 8x8 méretű blokkjában található információt a ***térbeli tartományból a frekvenciatartományba***

- Egyszerű analógia: Tekintsük 0 és 3 közötti számok következő listáját (*2, 3, 1, 2, 2, 0, 1, 1, 0, 1, 0, 0*). Tekintsük a következő kétlépéses transzformációt: (1.) rendezzük a listát (2.) számoljuk meg mindegyik előforduló szám darabszámát: (*4,4,3,1*). Ezzel a transzformációval elvesztettük a térbeli információt de megkaptuk a frekvencia-tartománybeli (gyakoriság) információt.
- Vannak olyan transzformációk, amelyek megtartják a térbeli információt is. Pl. Fourier transzformáció, DCT. Ez lehetővé teszi számunkra, hogy előre és hátra mozogjunk a térbeli és a frekvenciatartomány között.

JPEG: 2-D DCT

Kétdimenziós (2-D) transzformációhoz először soronként majd oszloponként alkalmazzuk az egydimenziós (1-D) DCT transzformációt: input f mátrix, output F mátrix.



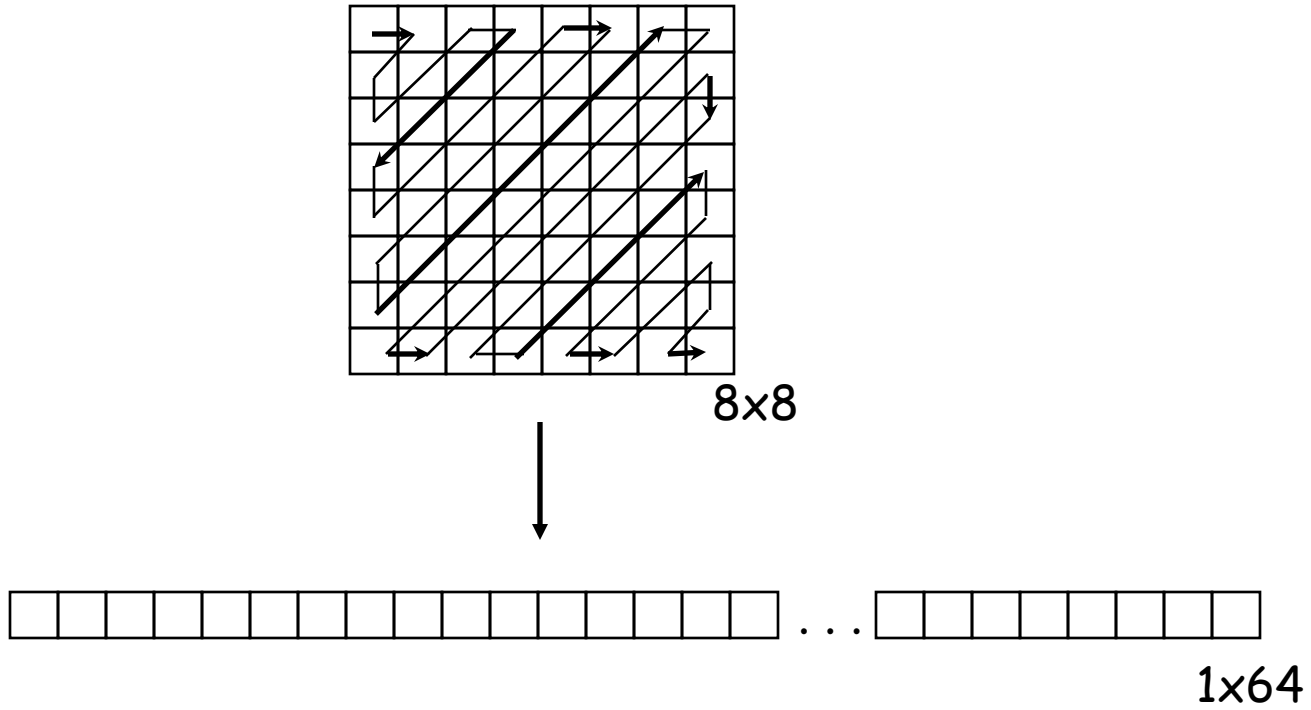
A 8x8-as transzformált négyzet (F) elemei a különböző felharmónikusoknak felelnek meg. F(0,0) neve ***DC komponens*** (bal felső sarokelem) a többi F(i,j) elem neve ***AC komponens.***

JPEG: kvantálás

- Kvantálással csökkentjük a bitek számát mintánként (→a tömörített kép méretét)
- Nagyobb frekvenciás együtthatókra nagyobb a kvantálási lépésköz, így ezen frekvenciákon nagyobb lesz a torzítás
- *A kvantálási hiba a veszteséges tömörítés fő forrása*
- A nem egyenletes kvantálást (frekvenciánként változó kvantálási lépésközt) kvantálási táblázattal definiáljuk
- A JPEG standard két default kvantálási táblázatot definiál: fényességre ill. a színekre

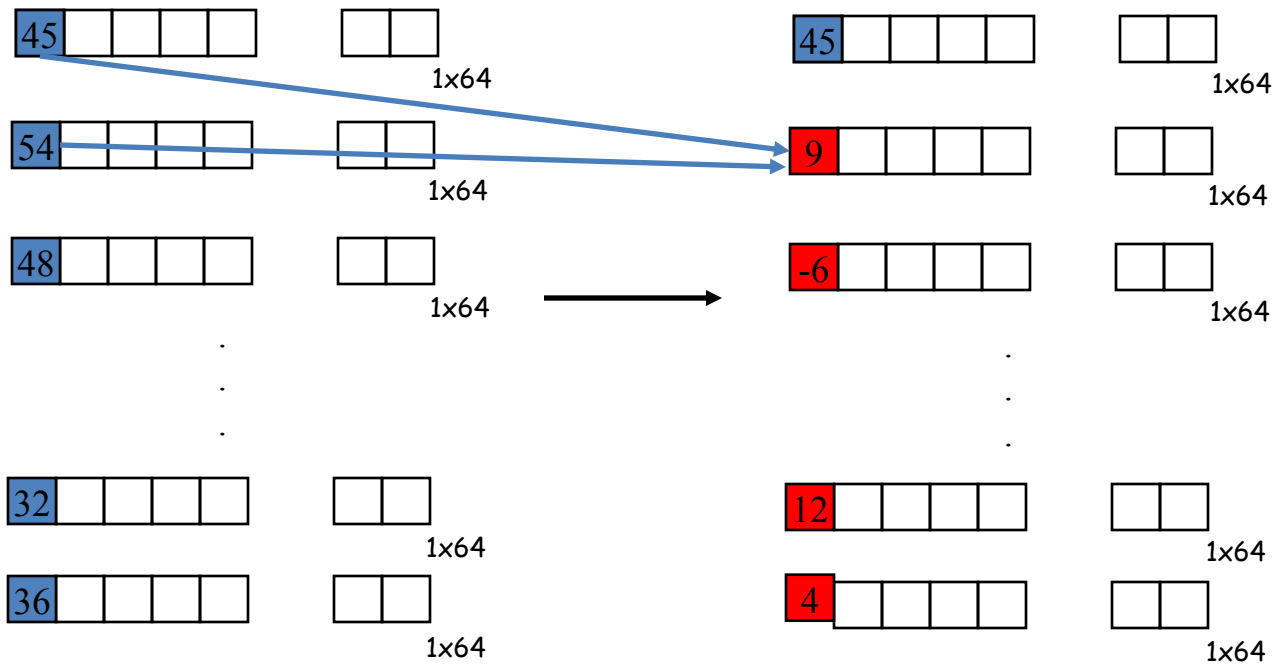
JPEG: cikk-cakk rendezés

- A cikk-cakk rendezés 8 x 8-as mátrixot a 1 x 64 vectorba transzformálja
- A cikk-cakk rendezés az alacsony frekvenciás együtthatókat csoportosítja az eredmény sorvektor elejére, a magas frekvenciás együtthatókat pedig a végére
- A magas frekvenciás komponensek durvábban kvantáltak, így a sorvektor vége felé túlnyomóan 0-k állnak



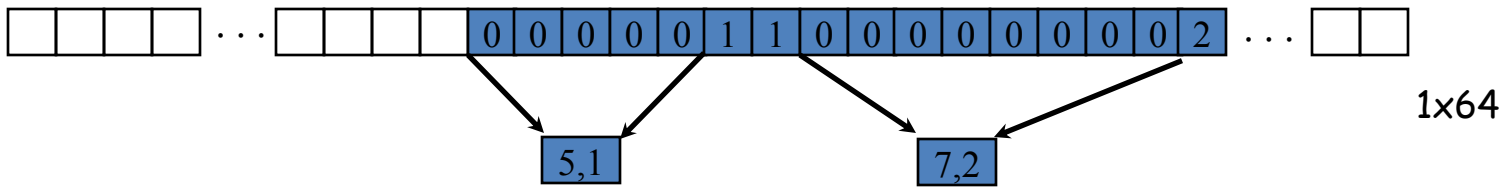
JPEG: DC komponensek DPCM kódolása

- A DC komponens értéke minden 8x8 blokkban nagy, és blokkonként változik, de gyakran megközelíti az előző blokk értékét.
- Differenciális Pulzus Kód Moduláció (DPCM) kódolja pillanatnyi és megelőző 8x8-as blokk differenciáját



JPEG: AC komponensek futamhossz kódolása

- A magas frekvenciás komponensek durvábban kvantáltak, így a sorvektor vége felé túlnyomóan 0-k állnak
- A 0 futamokat **(*skip*,*value*)** párokba kódoljuk, ahol *skip* a futambeli 0-k száma, *value* a futamot követő nemzérus komponens.
- (0,0) a blokk-záró pár.



JPEG: DC komponesek entrópia kódolása

DC komponeseket (**SIZE**,**Value**) párokba kódoljuk

– A **Value** kódolása az alábbi táblázat szerinti (egyes komplemens ábrázolás)

SIZE	Value	Code
0	0	---
1	-1,1	0,1
2	-3, -2, 2,3	00,01,10,11
3	-7,…, -4, 4,…, 7	000,…, 011, 100,…111
4	-15,…, -8, 8,…, 15	0000,…, 0111, 1000,…, 1111
.		.
.		.
11	-2047,…, -1024, 1024,… 2047	…

Size_and_Value Table

JPEG: DC komponesek entrópia kódolása

SIZE kódja a következő táblázat szerinti

SIZE	Code Length	Code
0	2	00
1	3	010
2	3	011
3	3	100
4	3	101
5	3	110
6	4	1110
7	5	11110
8	6	111110
9	7	1111110
10	8	11111110
11	9	111111110

Táblázat DC komponens SIZE paraméterére

Példa: Ha egy DC-komponens 40, és az előző DC-komponens 48. A különbség -8. Kódolva:

1010111

0111: –8 *Value* kódolása

(lásd Size_and_Value táblázatot)

101: Ugyanezen táblázatból kiolvasható Size=4. A kapcsolatos kód a bal oldali táblázat szerint 101.

JPEG: AC komponesek entrópia kódolása

- AC komponenseket (S1,S2) párokba kódoljuk:
  - **S1:** (**RunLength/SIZE**)
    - » **RunLength:** a 0 futam hossza [0..15]
    - » **SIZE:** a futamot követő nemzérus AC komponens ábrázolásához szükséges bitek darabszáma [0-A]
    - » (0,0) End_Of_Block (EOB)
    - » **S1** (Huffman) kódolása a jobboldali táblázat szerint
  - **S2: (Value)**
    - » **Value:** az AC komponens értéke (size_and_value táblázat szerint)

Run/ SIZE	Code Length	Code
0/0	4	1010
0/1	2	00
0/2	2	01
0/3	3	100
0/4	4	1011
0/5	5	11010
0/6	7	1111000
0/7	8	11111000
0/8	10	1111110110
0/9	16	1111111110000010
0/A	16	1111111110000011

Run/ SIZE	Code Length	Code
1/1	4	1100
1/2	5	11011
1/3	7	1111001
1/4	9	111110110
1/5	11	11111110110
1/6	16	1111111110000100
1/7	16	1111111110000101
1/8	16	1111111110000110
1/9	16	1111111110000111
1/A	16	1111111110001000
… 15/A	More	Such rows

JPEG: entrópia kódolás példa

40	12	0	0	0	0	0	0
10	-7	-4	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

…1111110111011010

Példa: Legyen a cikk-cakk rendezés outputja a következő:

-> 12,10, 1, -7, 2 db. zéró, -4, 56 db. zéró

12: kódolás: (0/4)12 → 10111100

1011: (0/4) kódolása (AC kódtábla)

1100: 12 kódolása (Size_and_Value táblázat).

10: (0/4)10 → 10111010

1: (0/1)1 → 001

-7: (0/3)-7 → 100000

2 zéró és -4: (2/3)-4 → 1111110111011

1111110111: (2/3) 10-bites kódja

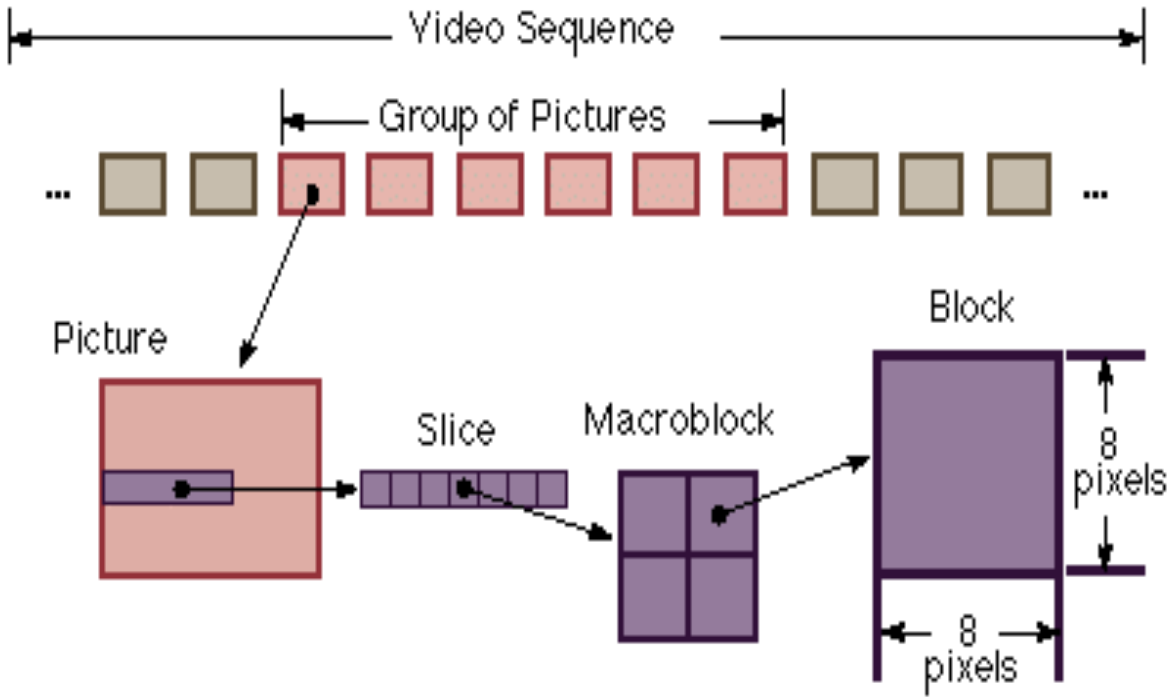
011: –4 kódolása (Size_and_Value táblázat)

56 db. zéró: (0,0) → 1010 (EOB)

MPEG

- MPEG-1 : a mozgóképek és hangok adathordozókon történő tárolására és visszakeresésére vonatkozó szabvány
- MPEG-2 : digitális televiziózás szabvány
- MPEG-4 : multimédia alkalmazások szabvány
- MPEG-7 : tartalomábrázolási szabvány az információ kereséséhez
- MPEG-21: metaadat-információkat kínál audio- és videofájlokhoz

MPEG: videófolyam hierarchia



Videoszekvencia
Képcsoport
Kép (I, P, B)
Sáv
Makroblokk (Y, Cb, Cr)
Blokk

MPEG: a tömörítés elve

- A tömörítés lényege a **mozgás-predikció.**
- A képcsoporton belüli tömörítés kezdete egy **predikciós kódolás**, ahol az egymás utáni képek hasonlóságát használjuk ki. A tömörítés makroblokk szinten történik. A makroblokk egy 16x16 pixeles képdarabon képzett 6 darab (8x8-as) blokkból áll. Ezen a 6 blokkon belül 1-1 blokkot használnak a színezéshez és 4 blokkot a fényerőhöz.
- Minden egyes makroblokkra keressük a legtöbb „hasonló” részletet az előző és / vagy a következő képeken; kiszámítjuk a ***tartalmi*** *különbséget* (**predikciós hibát**) és a ***térbeli*** *különbséget* is (**mozgásvektort**). Így a képsorozat mozgó részleteit az algoritmus követni tudja.
- Ezekből a differenciákból (ismét) 6 blokk keletkezik. Ezen blokkok mindegyikére a **JPEG** módszernél megismert DCT → kvantálás → futamhossz kódolás → Huffman kódolás kódolás-sort végezzük.

A valós eljárás egy kicsit összetettebb, mivel 4 különböző szabály alkalmazható:

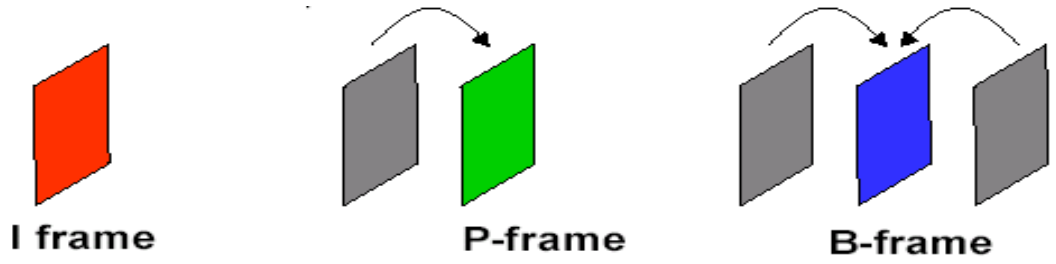
1) Tömörítés időbeli predikció nélkül (a térbeli redundancia JPEG-tömörítése kereten belül)
2) Tömörítés előre irányú predikcióval
3) Tömörítés hátra irányú predikcióval
4) Tömörítés előre és hátra irányú predikcióval

MPEG: képek típusai (I, P, B)

I-kép: nincs predikció a makroblokkok tömörítésénél; egy önálló kép JPEG tömörítése történik

P-kép: lehetnek predikció nélkül kódolva vagy predikcióval hátulról legközelebbi I vagy P képből

B-kép: P-kép kódolásához képesti különbség az, hogy előről is predikálhatunk legközelebbi I vagy P képből



Ellenőrző kérdések

- Miért hívjuk a veszteséges kódolást veszteségesnek?
- Nevezzen meg veszteséges audio kódolás alkalmazásokat!
- Nevezzen meg veszteséges kép kódolás alkalmazásokat!
- Nevezzen meg veszteséges video kódolás alkalmazásokat!
- Hogyan definiáljuk a torzítást veszteséges audio kódolásnál?
- Mi a JPEG konstrukció tapasztalati háttere?
- Mi a DCT transzformáció célja JPEG kódolásnál?
- Mi a veszteség fő forrása JPEG kódolásnál?
- Milyen tapasztalatot használ a nem-egyenletes JPEG kvantálás?
- Mi a futamhossz kódoló tömörítés ötlete JPEG-nél?
- Mi a Huffman-kódolás célja JPEG-nél?
- Mik a redundancia fő forrásai video esetén? Hogyan használja ezt ki az MPEG kódolás?