Adatrejtés / Adatrejtés képekben

BME - TMIT

VITMA378 - Médiabiztonság

feher.gabor@tmit.bme.hu

A legjelentéktelenebb bit (LSB) szerepe

- Least Significant Bit (LSB) felülírása
  - A legjelentéktelenebb bitek változtatása kép, hang, videó média esetén nem feltűnő
    - Nem LSB megváltoztatása azonban már felismerhetővé teheti az adatrejtést
  - Zaj-szerűen kell elhelyezni a rejtett adatokat (mintha AD/DA átalakítás lenne)
  - A megváltoztatandó pixelek kiválasztása
    - Sorban, álvéletlen bejárás, bizonyos pixelek (pl. élek mentén)

- Az egyik leggyakrabban használt eljárás
  - Az LSB értéke az elrejteni kívánt bit értéke lesz – vak módszer
    - LSB -> 0 : rejtett tartalom 0 bit
    - LSB -> 1: rejtett tartalom 1 bit

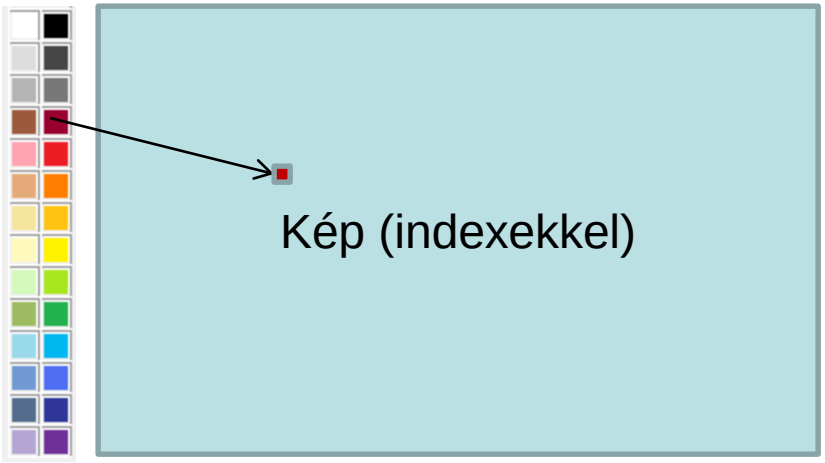
Tömörítetlen képek

- A tömörítetlen / nem veszteségesen tömörített képek forrásai:
  - Magas minőségi követelmények (orvosok, fényképészek)
  - TIFF, BMP tárolás

- Az LSB módszer közvetlen alkalmazása a pixeleken
  - Az információt a pixelek színének LSB bitjeibe rejtem
  - Veszteséges tömörítés hatására a rejtett információ sérülhet, hiszen ez érinti az LSB biteket

Paletta alapú képek

- A paletta alapú képek leggyakoribb forrásai:
  - A színek számának csökkentése
  - Számítógép által generált képek (fraktálok, képregények,…)
  - BMP, GIF fájlok



Rejtés palettás képekbe

- Az indexeken közvetlenül nem alkalmazható az LSB adatrejtés
  - Mert a palettában távoli színek is egymás mellé kerülhetnek
  - Palettából kimutató indexek jöhetnek létre

- Lehetőségek:
  - Adat rejtése a palettába
  - Adat rejtése a képpontok közé

Adatrejtés a palettában

- A paletta permutálása
  - A kép nem változik
  - Limitált kapacitás
  - Gyanúsan kevert paletta

- LSB kódolás a palettában
  - Limitált kapacitás (gyakran max. 3x256 bit)

- Az elrejtett adat mennyisége független a kép méretétől, csak a palettától függ

Adatrejtés a képpontokban

- Csökkentett színmélység + kiterjesztés
  - A színmélység megfelezése, majd közeli színekkel megduplázása
  - A rejtett adatot a paletta közeli színei hordozzák (1 bit)
  - A palettát adott esetben 32 színre is lehet redukálni, majd 256 színre bővíteni (3 hasznos bit)

Paletta méretének csökkentése

Lena.bmp

256 szín

32 szín

Lena Söderberg

1972. nov. Playboy

Adatrejtés a képpontokba (LSB)

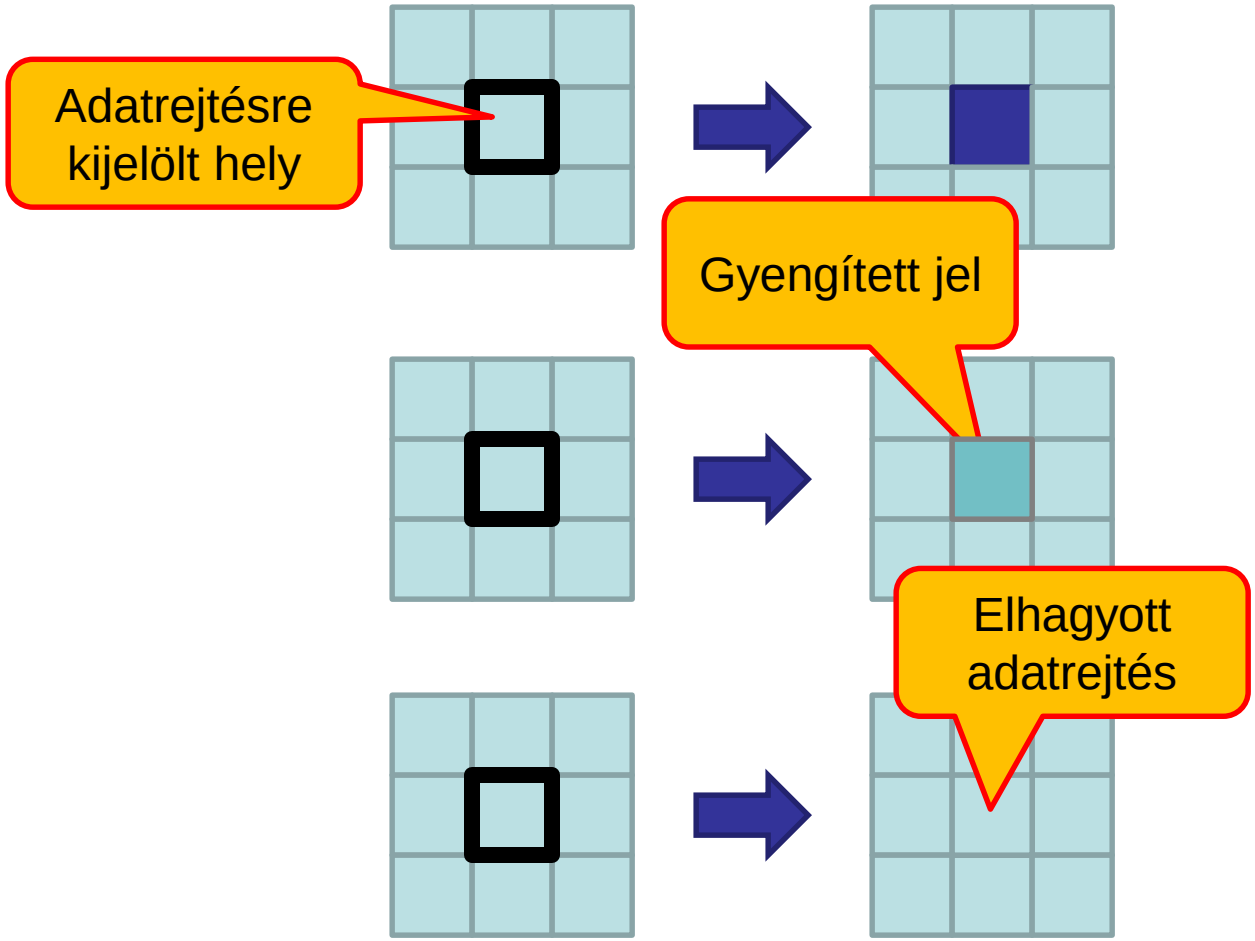
- Paritásos kódolás
  - A kiválasztott képpontok indexének cseréje egy megfelelő paritású színre a palettában
  - $d^2 = (R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2$
  - Csak akkor működik, ha vannak közeli színek

Adaptív adatrejtés

- Nem adaptív adatrejtés
  - A kijelölt (stego key) adatrejtő képpontok függetlenek a képi tartalomtól
  - Feltűnően eltérő képpontok jöhetnek létre
- Adaptív adatrejtés
  - A kijelölt adatrejtő képpontok figyelembe veszik környezetüket. Pl.:
    - Homogén (egyszínű) tartományok kerülése
    - Feliratok kerülése
    - Nagy szórású helyek választása
  - Adatrejtés gyengítése vagy elhagyása

Adaptív adatrejtés (folyt..)

- Nem adaptív
- Adaptív #1
- Adaptív #2

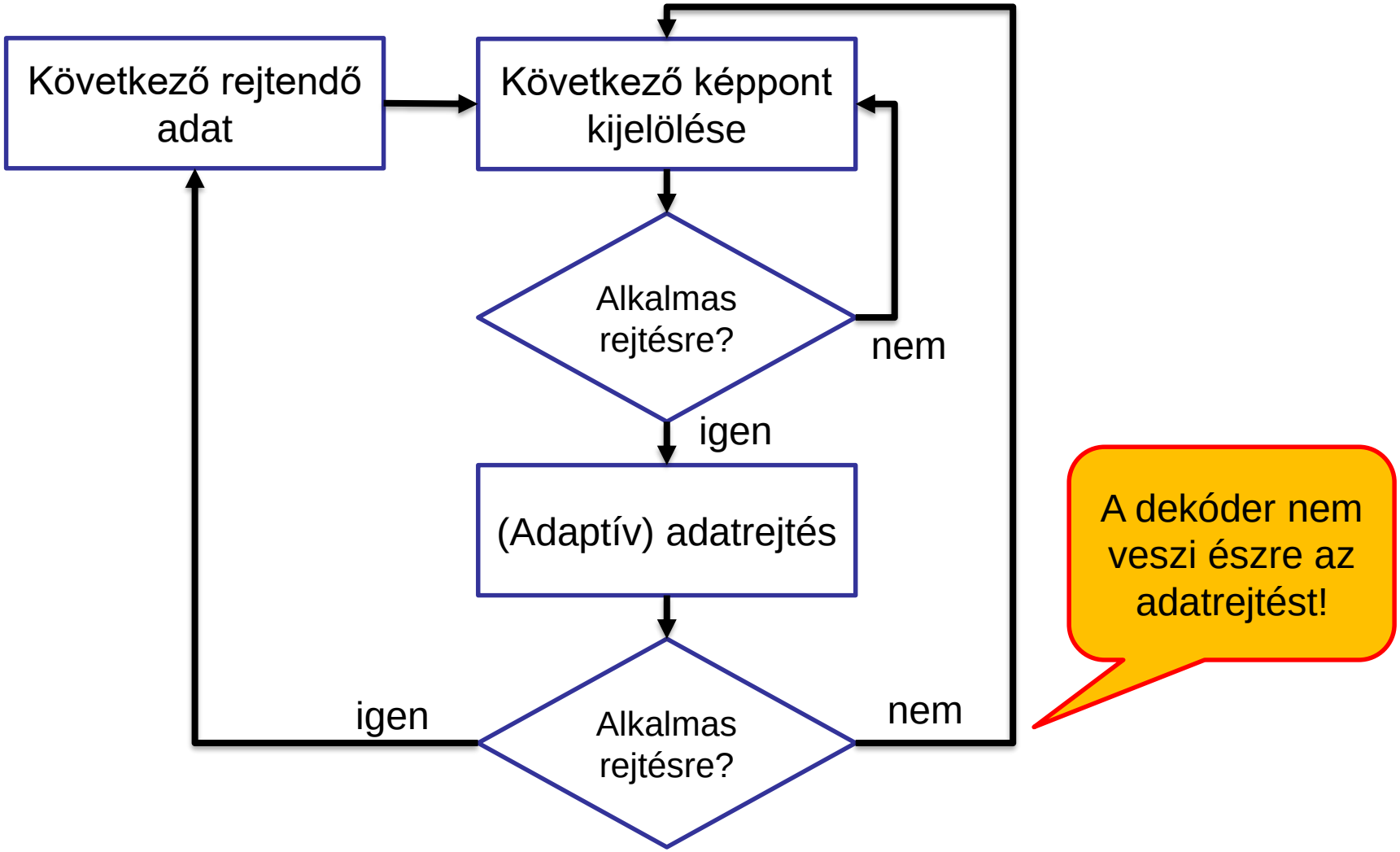


Adaptív adatrejtés (folyt..)

- Vak adatrejtés esetén szükséges, hogy a rejtett információs helyeket a dekódolásnál is azonosítani tudjuk!
  - A rejtés helyét a mindkét oldalon egy álvéletlenszám generátor adja. Dekódolásnál tudnunk kell, hova nem rejtettünk adatot.

- Egy lehetséges megoldás:
  - A kép felosztása kis (pl. 3x3) területekre
  - A blokk véletlen (stego key) kijelölése és ellenőrzése: ***jó*** blokkok és ***rossz*** blokkok
  - Ha ***jó*** a blokk, akkor itt adatrejtés
  - Ha a blokk adatrejtés hatására „elromlik” akkor ismétlés a következő blokkban is
  - Az adat visszaállítása csak a ***jó*** blokkok alapján

Adaptív adatrejtés (folyt..)



Adaptív adatrejtés (folyt..)

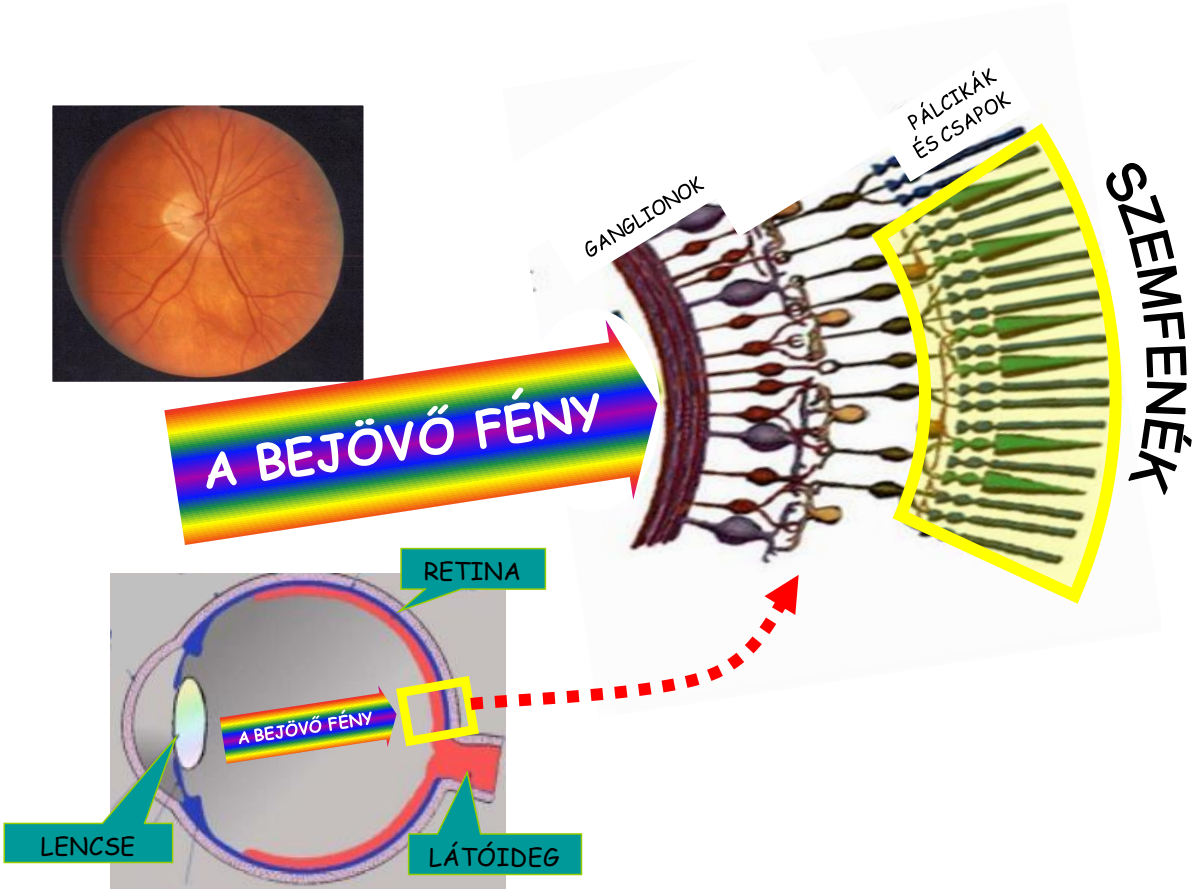
- Nem egyszerű adaptív megoldásokat találni, nagyon bonyolult érzékelés kell hozzá.
- Gép és ember eltérő képességei
  - Pl.: az i –n a pont egyszerű egy embernek, de nem az egy gépnek
  - A gép könnyebben kimutat statisztikai változásokat, amiket az ember nem vesz észre

Veszteségesen tömörített képek

- Veszteséges tömörítés
  - A tömörített kép nem egyezik az eredeti képpel, van ami hiányzik, de (emiatt) a mérete kisebb
    - Azok a részek, amelyek az emberi szem számára nem láthatóak, nem szükségesek a képen

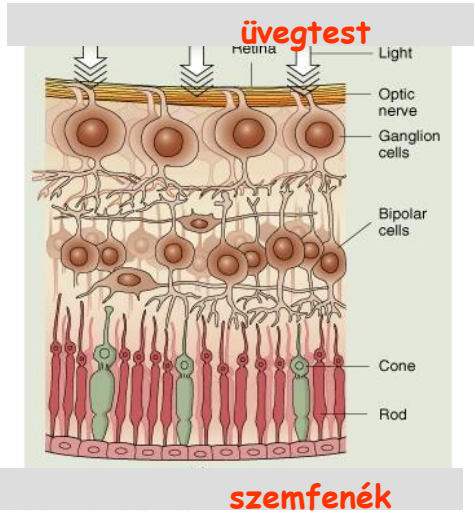
Emberi látás

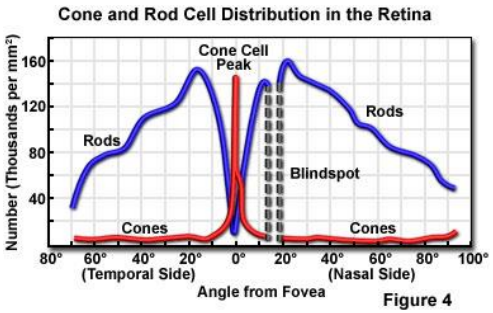
- Szem felépítése
  - csapok
  - pálcikák



Emberi látás (folyt.)

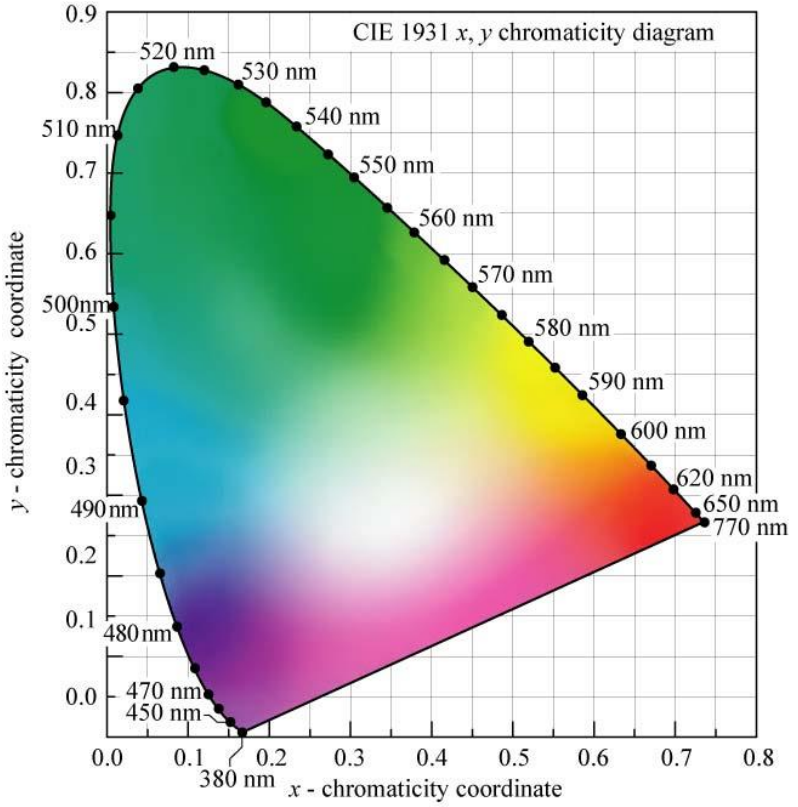
- Csapok (cone)
  - ~7 millió csap. Gyenge fényben nem működik. Felelős a színlátásért. A periférián nincsenek
    - R csap: ibolya szín (~1 mill.)
    - K csap: zöld szín (~2 mill.)
    - H csap: sárga szín (~4 mill.)
- Pálcikák (rod)
  - ~120 millió pálcika. Kiváló fényérzékenység. Nem vesz részt a színlátásban.
- A fizikai méretekből/távolságokból következtethető a felbontási képesség is





Színek látása - színtér

- CIE
  - International Commission on Illumination (CIE)
  - 1931 –ben az első matematikai leírása a színlátásnak



Tömörített képek - JPEG

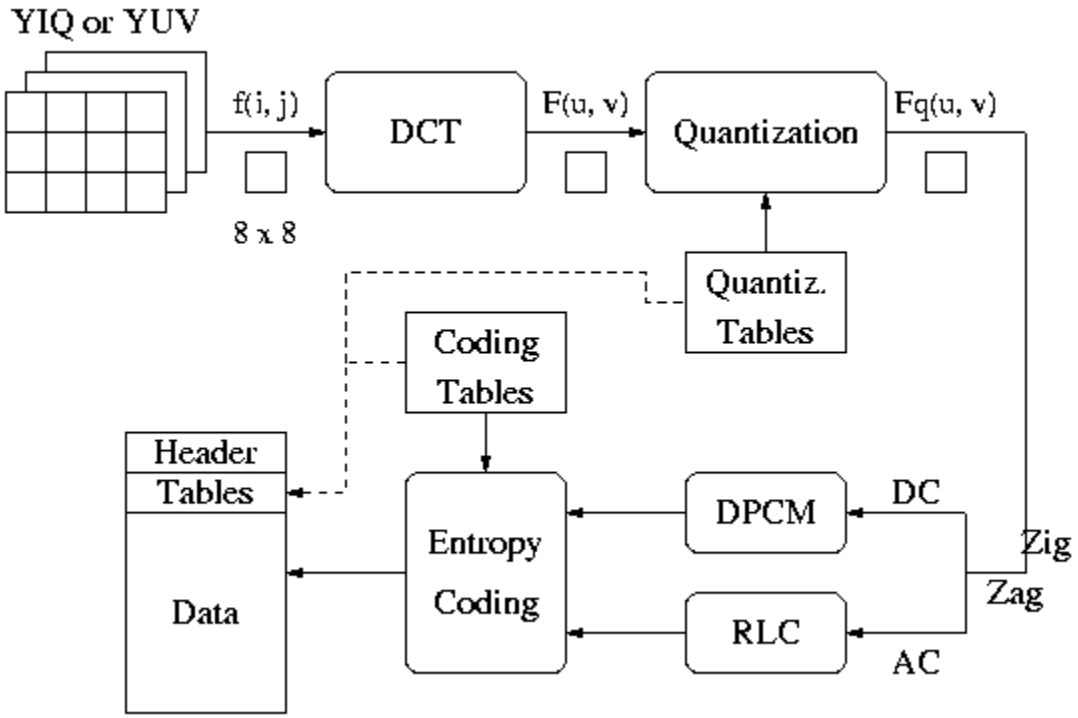
- JPEG
  - Joint Photographic Expert Group (1986)
  - CCITT (ITU) és ISO szabvány – 1992/94

- Tömörítés alapja:
  - Kvantálás
  - Kódolás (run-length, Huffman)
  - (Diszkrét koszinusz transzformáció (DCT))

Fontosság és méret kapcsolata

- Veszteséges (10:1 – 100:1) és veszteségmentes (1.5:1 - 2:1) tömörítés

JPEG kódolás menete



1. RGB -> YUV

- RGB: Vörös, zöld, kék
  - Egyenletesen kvantálva (általában 8 bit)

- Luminancia (Y - fényesség) és krominancia (U, V) összetevők
  - Az emberi szem kevésbé érzékeny a krominancia összetevőkre
  - Valójában relatív luminancia (fehérhez képest)

- Konverzió

$$\begin{bmatrix} Y' \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

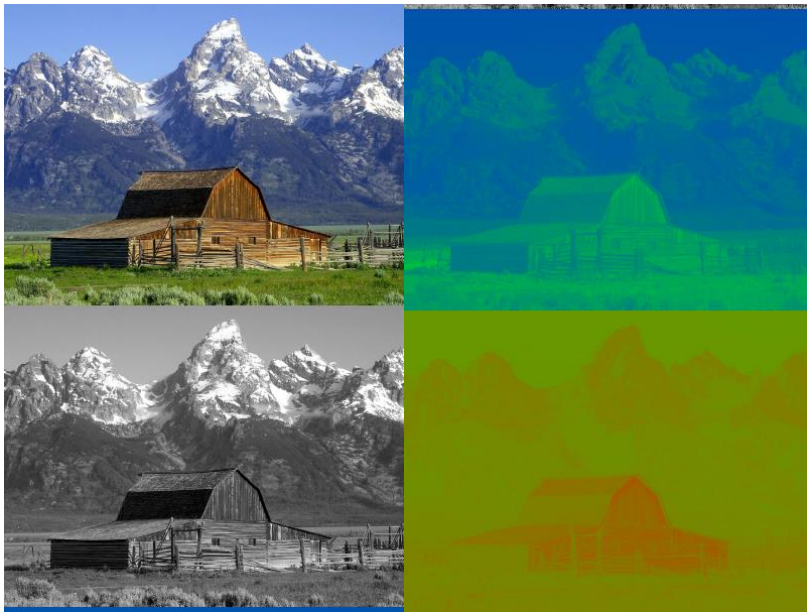
SDTV

$$\begin{bmatrix} Y' \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.2126 & 0.7152 & 0.0722 \\ -0.09991 & -0.33609 & 0.436 \\ 0.615 & -0.55861 & -0.05639 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

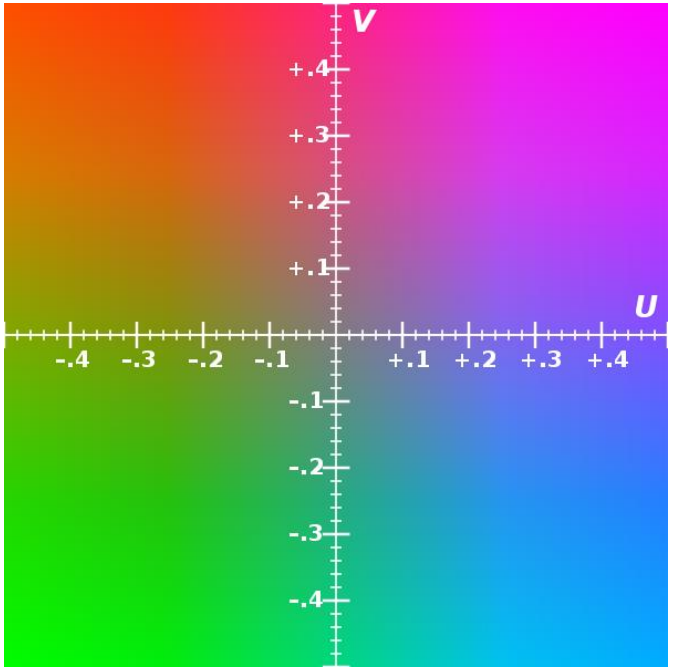
HDTV

YUV

- Luminancia (Y) és krominancia (UV)



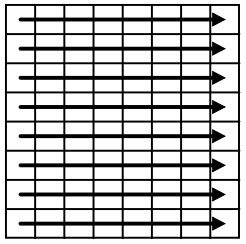
	U
Y	V

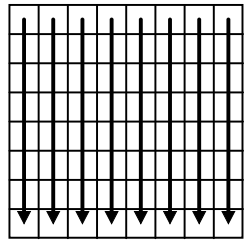


Y=0.5

2. DCT

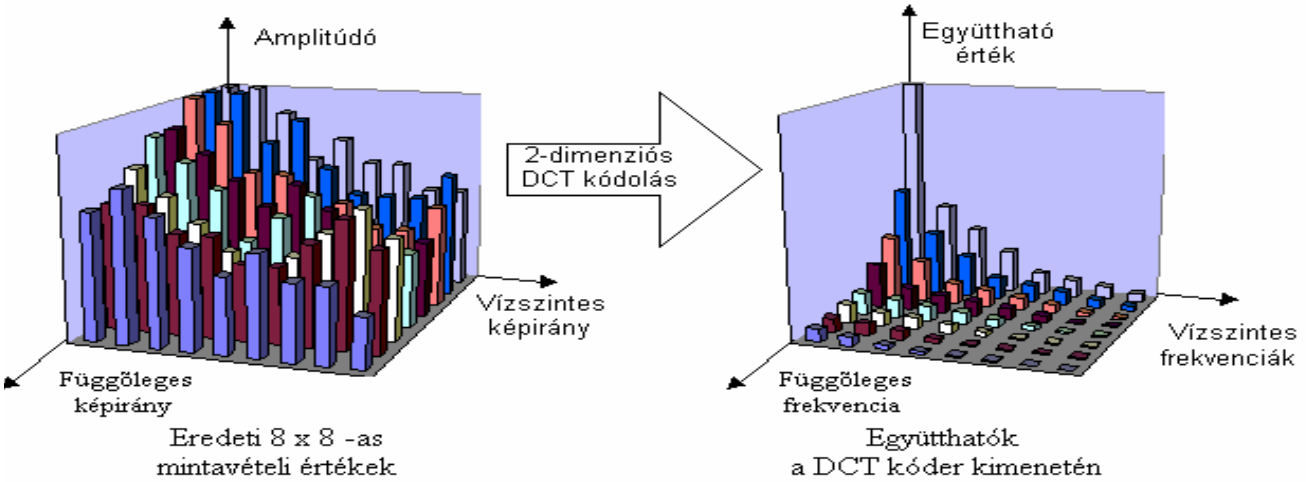
- Diszkrét koszinusz transzformáció (DCT)
- 8x8 –as blokkok a bemeneten
  - Egyszerűbb, gyors kódolás
  - Ellenben „blokkosodás”
- Leképezés a frekvencia-tartományba
  - $DCT_i = \sqrt{\frac{C}{N}} \sum_{j=1}^{N} \cos\left(\frac{2j+1}{2N} i\pi\right) x_j$ C=1, ha i=1; C=2, ha i>1
  - 2 dimenziós
  - DC komponens (bal felső)
  - AC komponensek





2. DCT II.

- A kép egy blokkja csak kicsi átmeneteket tartalmaz
  - Alacsony frekvenciák jelentősebbek
  - Az emberi szem nem látja jól a magas frekvenciát
    - Magas frekvenciákat nem szükséges jól ábrázolni

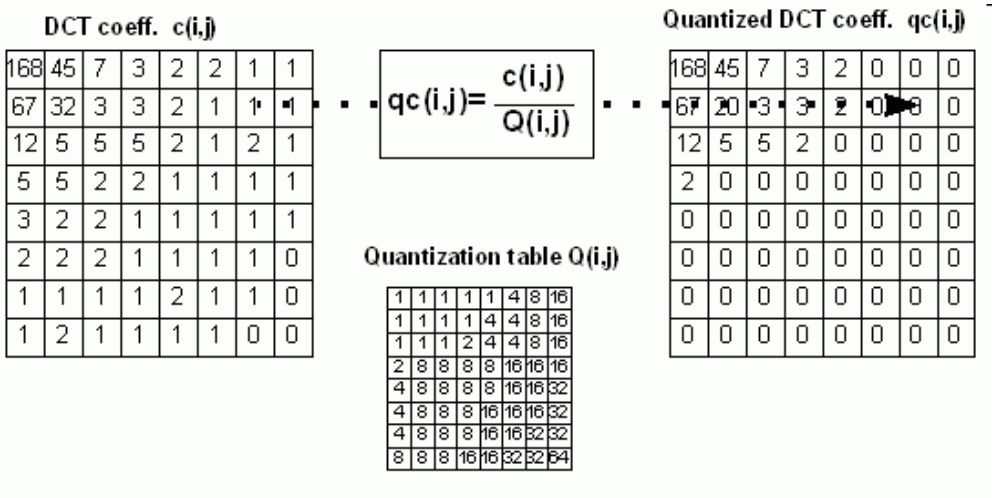


DCT – Mátrix műveletekkel

- A DCT transzformáció mátrix műveletekkel is elvégezhető
- D = T·M ·T’
  - Az M mátrix a 8x8 képtartomány -128 – 127 tartományban.
  - $$T_{i,j} = \begin{cases} \frac{1}{\sqrt{N}} & \text{if } i = 0 \\ \sqrt{\frac{2}{N}} \cos\left[\frac{(2j+1)i\pi}{2N}\right] & \text{if } i > 0 \end{cases}$$ *(N=8 a 8x8 -ra)*
  - T ortogonális, T inverze T’
  - M értéke a pixel értékek mínusz 128

3. Kvantálás

- Egészen idáig veszteségmentes volt az átalakítás (színtér és DCT)
- Meghatározzuk az egyes frekvenciák súlyát
  - Ezzel irányítható a tömörítés mértéke
  - Az emberi látáshoz van igazítva



JPEG minőség

- A JPEG minőséget szokás 1-100 –ig leírni. A Q_{50} kvantálási tábla van leírva (szabvány), a többit ebből lehet számolni.
- 50-nél jobb minőség esetén a táblázat elemeit (100-Q)/50 –nel szorozzuk.
- 50-nél rosszabb minőség esetén a táblázat elemeit (50/Q) –val szorozzuk.

Luminancia kvantáló mátrix

$$Q_{50} = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

JPEG minőségek

- Q: 1 (144:1) – 10 (46:1) - 50 (15:1) -100 (2.6:1)

4. Kódolás

- Veszteségmentes tömörítés
  - A DC komponenseket DPCM kódolással lehet tárolni (csak a különbséget tárolom)
  - Az AC komponenseknél RLC (futamhossz kódolás), de a hatékonyság érdekében cikk-cakkban vannak az adatok!
  - A kapott kódokat még entrópia (pl. Huffman) kódolhatjuk is

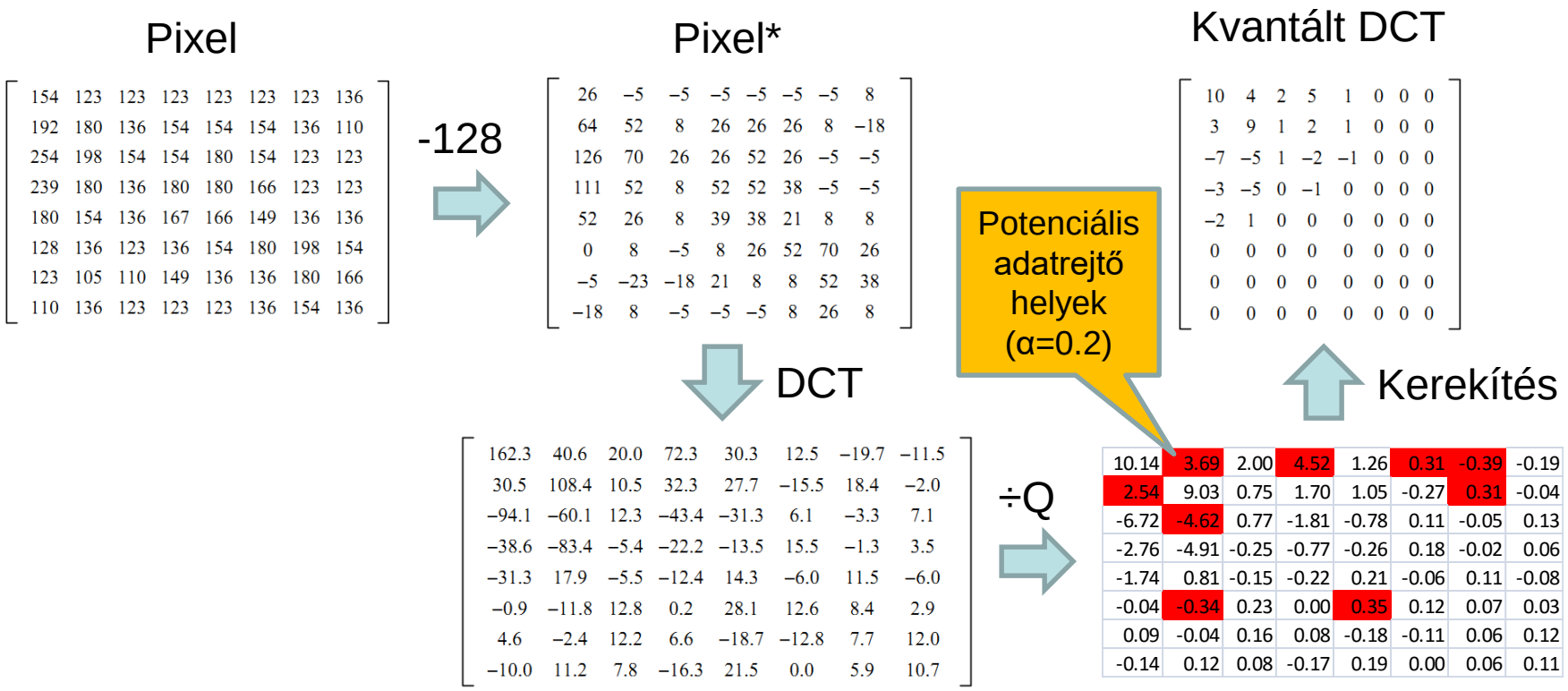
Adatrejtés JPEG képben

JPEG adatrejtés kerekítéssel

- Az adatokat a kvantálás kerekítésébe rejtem el, ahol a kerekítendő érték közel van <egész>.5 –höz
  - 1: Felfelé kerekítek, 0: lefelé kerekítek
  - Nem feltűnő változás

- Adatrejtés lépései:
  - 1. Az elrejtendő adat hibajavító kódolása
  - 2. A kép kódolása során a megfelelő helyek felderítése (α paraméter a közelséghez)
  - 3. A kerekítések igazítása az elrejtendő adatok szerint (redundánsan)

JPEG adatrejtés kerekítéssel (folyt.)

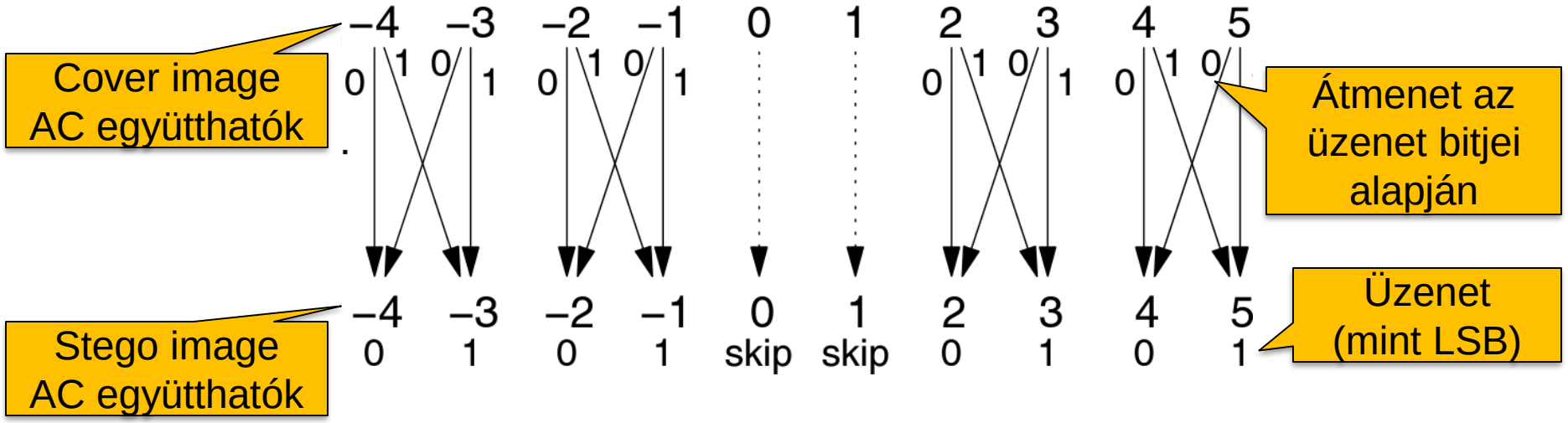


JPEG adatrejtés kerekítéssel (folyt.)

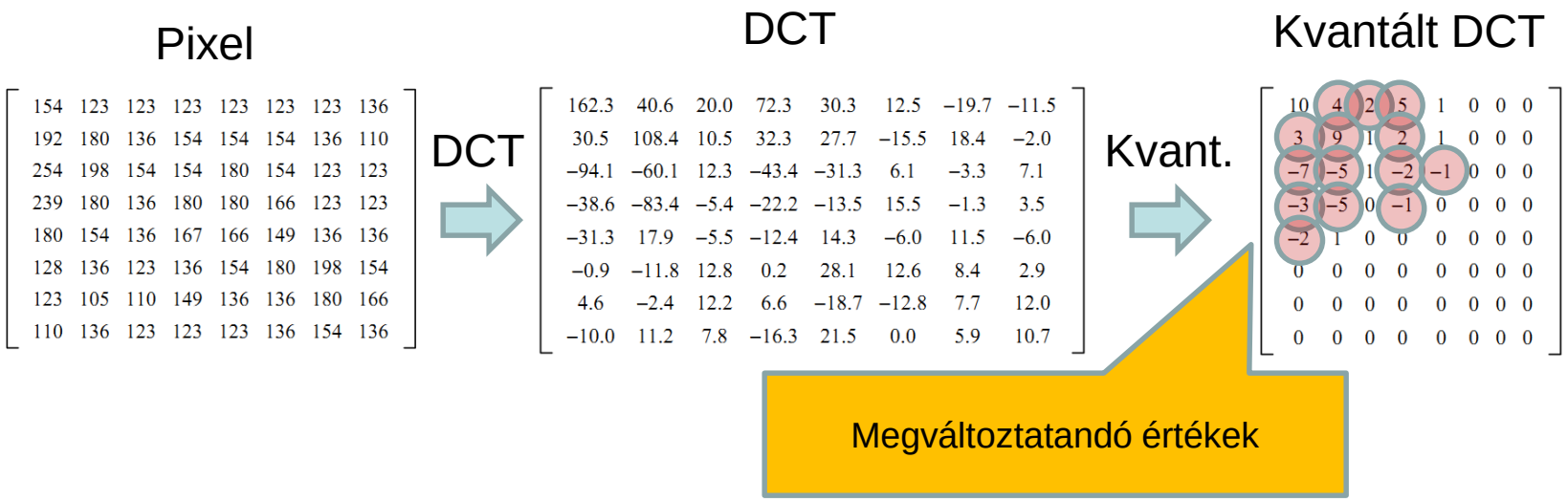
- A rejtett információ visszanyerése:
  - Nem vak: Szükség van az eredeti képre! Hiszen csak így tudom meg, hol volt kerekítés
- Rejtett adat visszanyerésének lépései:
  - 1. Az adatrejtési helyek felderítése az eredeti kép alapján
  - 2. Az ott lévő kerekítésekből a rejtett adatok visszanyerése

JPEG adatrejtés - JSteg

- Algoritmus működése
  - Adatrejtés a kvantált DCT együtthatókba
  - A nem 0 és nem 1 AC együtthatók LSB igazítása a rejtett üzenet alapján
  - Vak adatrejtés: A kvantált DCT együtthatók LSB vizsgálatával megállapítható az elrejtett adat



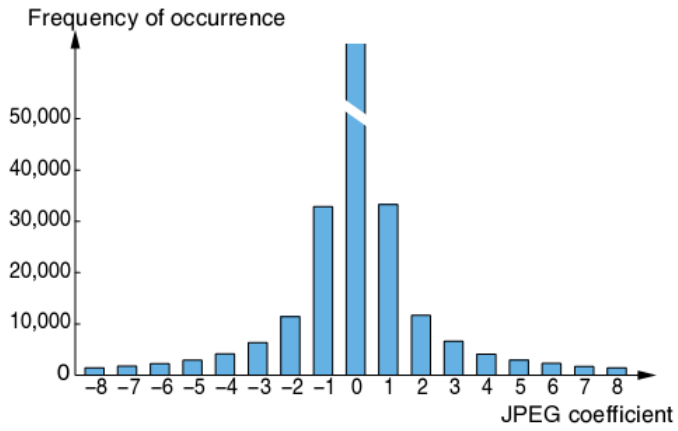
JSteg (folyt.)


Pixel
154 123 123 123 123 123 123 136
192 180 136 154 154 154 136 110
254 198 154 154 180 154 123 123
239 180 136 180 180 166 123 123
180 154 136 167 166 149 136 136
128 136 123 136 154 180 198 154
123 105 110 149 136 136 180 166
110 136 123 123 123 136 154 136
DCT
DCT
162.3 40.6 20.0 72.3 30.3 12.5 −19.7 −11.5
30.5 108.4 10.5 32.3 27.7 −15.5 18.4 −2.0
−94.1 −60.1 12.3 −43.4 −31.3 6.1 −3.3 7.1
−38.6 −83.4 −5.4 −22.2 −13.5 15.5 −1.3 3.5
−31.3 17.9 −5.5 −12.4 14.3 −6.0 11.5 −6.0
−0.9 −11.8 12.8 0.2 28.1 12.6 8.4 2.9
4.6 −2.4 12.2 6.6 −18.7 −12.8 7.7 12.0
−10.0 11.2 7.8 −16.3 21.5 0.0 5.9 10.7
Kvant.
Kvantált DCT
10 4 2 5 1 0 0 0
3 9 1 2 1 0 0 0
−7 −5 1 −2 −1 0 0 0
−3 −5 0 −1 0 0 0 0
−2 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
Megváltoztatandó értékek

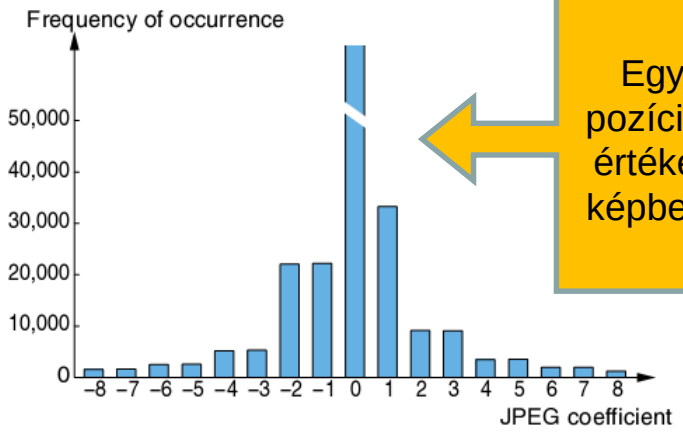
JSteg detektálása

- A JSteg adatrejtés átmeneteinek hatására az AC együtthatók aránya megváltozik
  - Az eredeti szimmetria megbomlik
    - AC párok kialakulása (Feltételezve, hogy a 0 és 1 egyenlő mértékben van a rejtendő szövegben)
    - Pozitív és negatív oldal nem egyformám változik

Cover kép

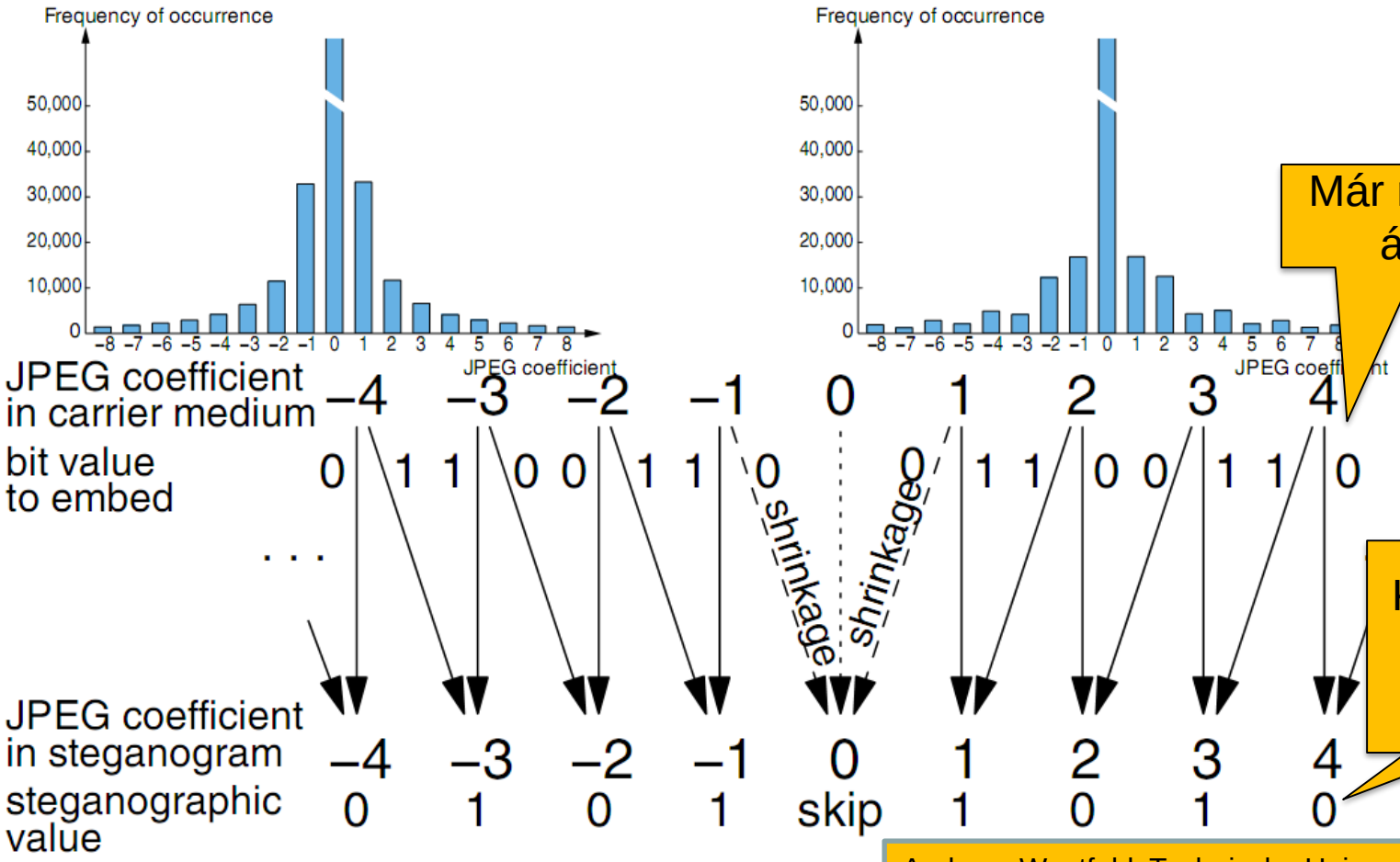


Stego kép

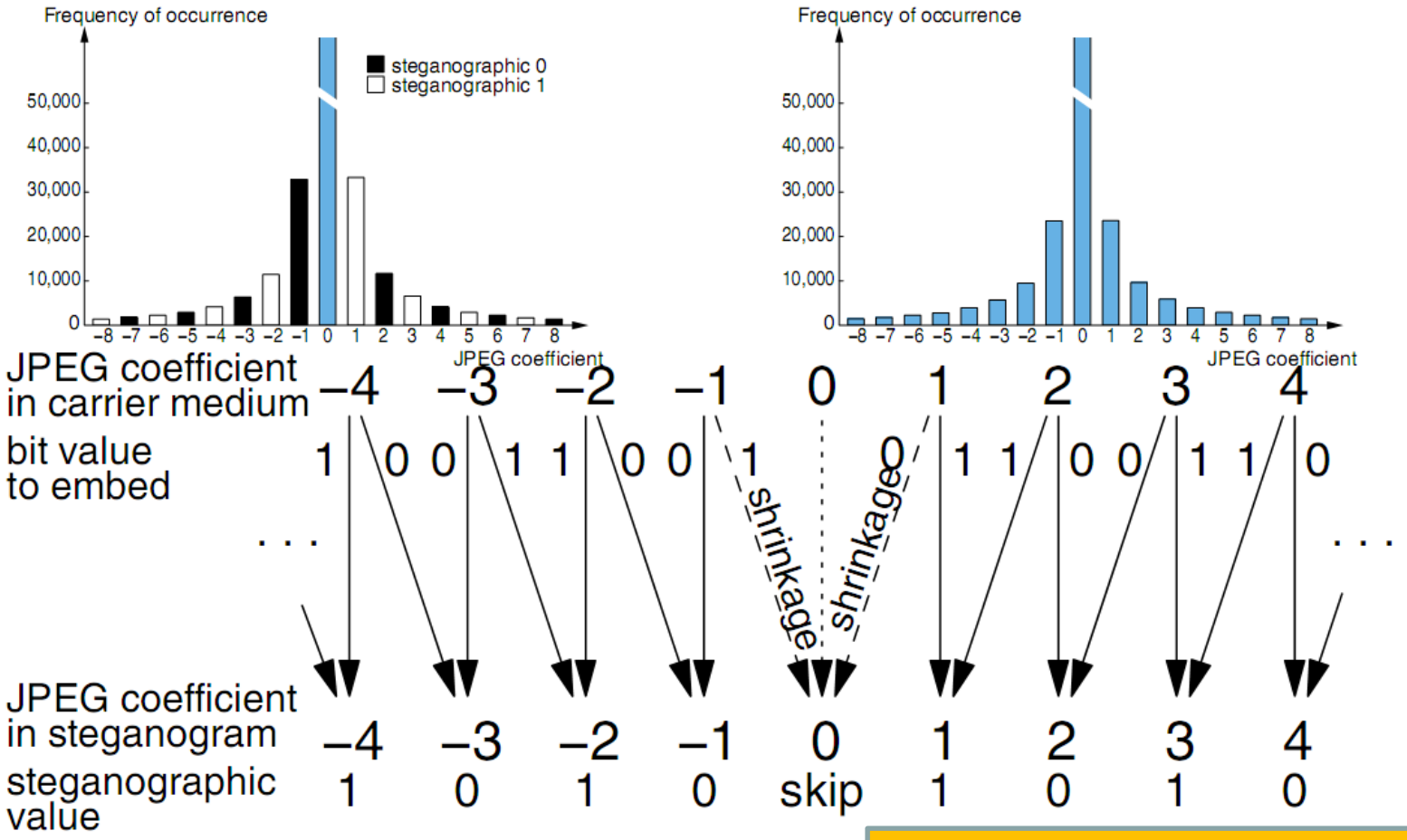


Egy adott AC pozíció (pl. $AC_{3,4}$) értékei az egész képben vizsgálva

F3 adatrejtés



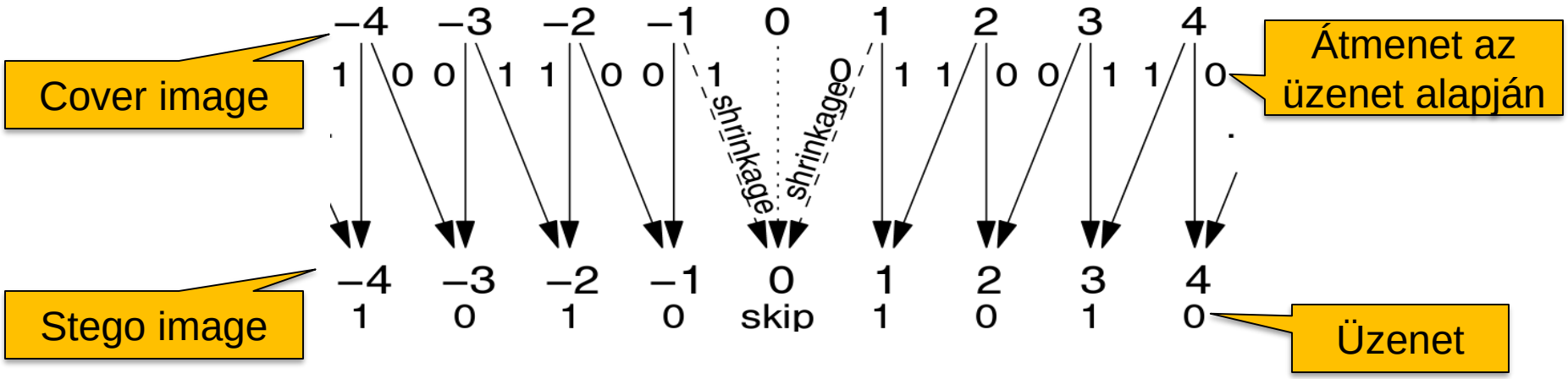
F4 adatrejtés



Andreas Westfeld, Technische Universitat Dresder

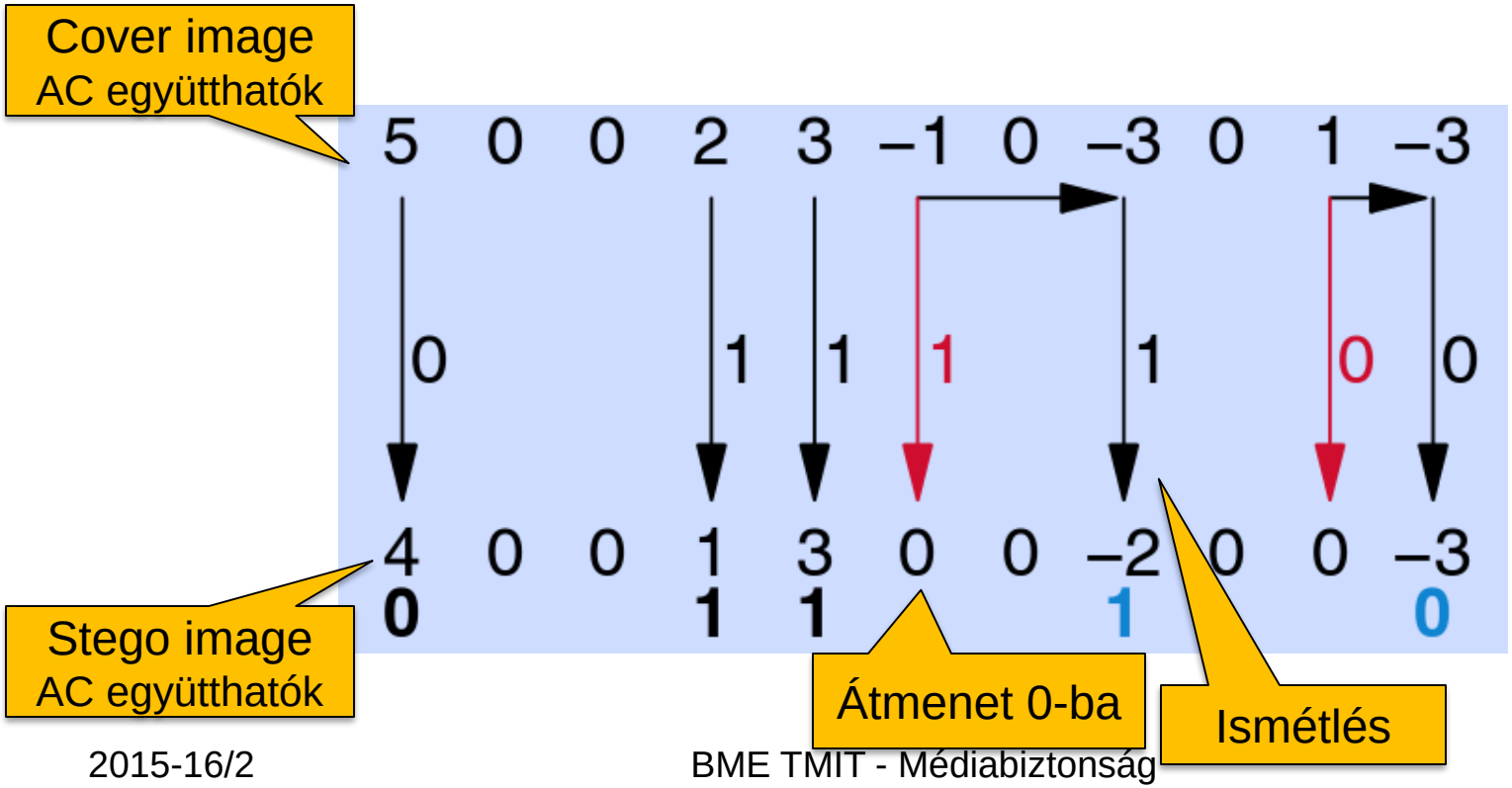
JPEG adatrejtés - F5 algoritmus

- F5: Pfitzmann and Westfeld 2001
- Algoritmus működése
  - Adatrejtés a DCT együtthatókba
    - A nem nulla AC együtthatók igazítása a rejtett üzenet szerint
      - +/-1 vagy 0 változás az elrejtendő bit szerint
    - Ha kinullázódik, akkor ismétlés a következőben pozícióban is (a detektálás miatt elengedhetetlen)



F5 algoritmus - példa

- AC átmenetek
  - Elrejtendő üzenet: 01110



F5 algoritmus – Mátrix kódolás

- Mátrix kódolás
  - k bit rejtése 1 bit változtatásával 2^k-1 bit hosszú AC csoportokban
    - k értéke az üzenettel együtt van rejtve, a legnagyobb k értéket választjuk
    - Csak egyetlen AC bit változik a 2^k-1 bit hosszú csoportban!
  - Az adatrejtés kapacitása (sebessége) csökken
  - Kevesebb bit változik, ezért nő az észrevétlenség, azaz biztonságosabb
- Szteganografikus kapacitás
  - JPEG kép13%-a (magasnak számít)

F5 algoritmus – Mátrix kódolás (példa)

- Tegyük fel, hogy 2 bitet akarunk megváltoztatni (x_1 és x_2) és 3 helyen változtathatunk (a_1, a_2 és a_3)

k = 2

$x_1 = a_1 \oplus a_3,\ x_2 = a_2 \oplus a_3$ — Minden rendben

$x_1 \neq a_1 \oplus a_3,\ x_2 = a_2 \oplus a_3$ — a_1 cseréje kell

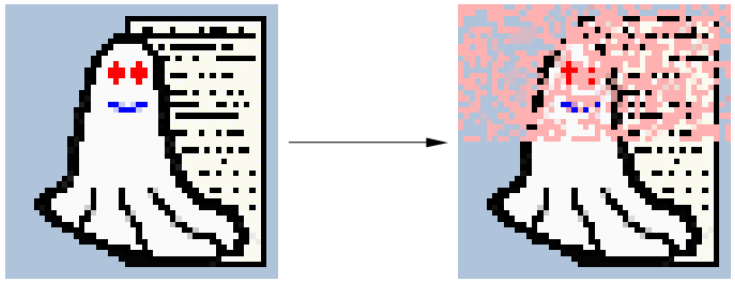
$x_1 = a_1 \oplus a_3,\ x_2 \neq a_2 \oplus a_3$ — a_2 cseréje kell

$x_1 \neq a_1 \oplus a_3,\ x_2 \neq a_2 \oplus a_3$ — a_3 cseréje kell

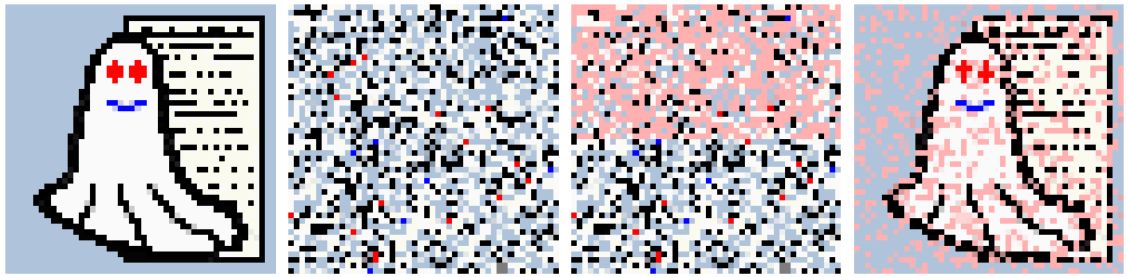
F5 algoritmus - Straddling

- Az elrejtett információ elszóródik a képben
  - A képet először permutáljuk (álvéletlen)
  - A permutált képben folyamatosan rejtünk
  - A képet visszaállítjuk (visszapermutáljuk)

Folyamatos adatrejtés

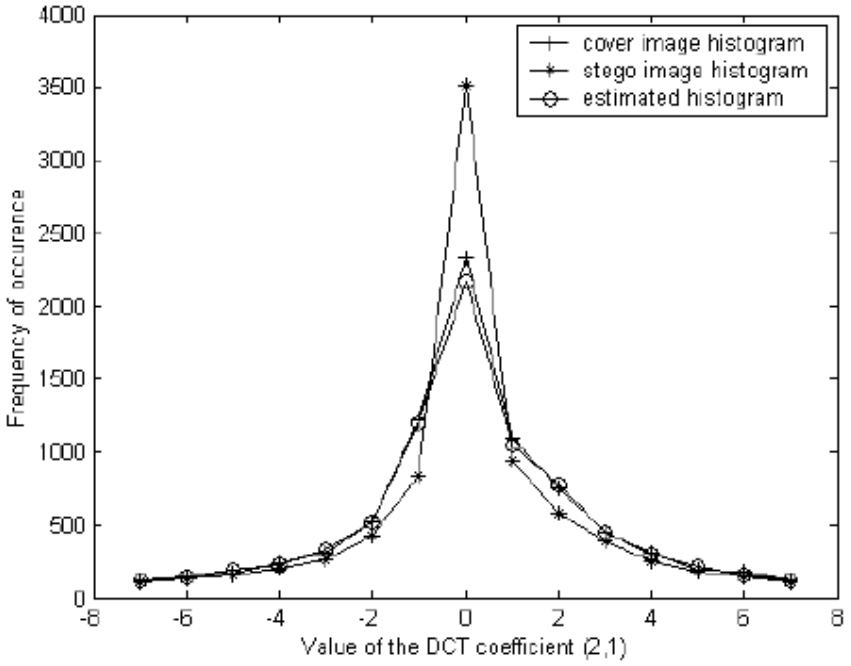


Adatrejtés permutációval



F5 felismerése

- Az eredeti kép AC DCT együtthatóinak becslése
  1. Kitömörítés
  2. Blokkhatárok elmosása (3x3 blur)
  3. 4 pixeles oszlopvágás
  4. JPEG tömörítés
  5. AC gyakoriság kiszámítása (Csak alacsony frekvenciák)
- A rejtett üzenet hossza is becsülhető



JPEG adatrejtés - OutGuess

- OutGuess – Provos 2001
- Minimális adatrejtés
- Két fordulós adatrejtés
  1. Adatrejtés véletlen bejárással, F5 alapján. +/-1 vagy 0 változtatás.
  2. A fel nem használt AC együtthatók segítségével az AC gyakoriság diagram visszaállítása

- Mivel az AC gyakoriság diagram nem változik, ezért az adatrejtés ennek vizsgálatával felismerhetetlen

OutGuess felismerése

- A képtartománybeli blokkosodás mértéke függ az Outguess adatrejtéstől
  - A blokkosodás változása eltér abban az esetben, ha már történt OutGuess adatrejtés
  - Módszer:
    1. Kitömörítés, blokkosodás kiszámítása
    2. A maximális mértékű üzenet elrejtése és a blokkosodás ismételt kiszámítása
    3. A kép 4 oszlopos vágása és a blokkosodások kiszámítása
    4. A két kép közötti különbségből az üzenet hossz megállapítható

Szteganalízis

Szteganalízis

- Szetganográfiai algoritmus erőssége
  - Chandramouli és Memon (2002)
    - Téves felismerés valószínűsége:
      P_{FP} = P(felismerés | nincs üzenet)
    - Valódi felismerés valószínűsége:
      P_{TP} = P(felismerés | rejtett üzenet)
  - Egy algoritmus γ erős, ha $| P_{FP} - P_{TP} | \leq \gamma$
  - Ha γ = 0, akkor tökéletes az algoritmus

Ha éppen annyira bizonyos, hogy rejtett, mint amennyire bizonyos, hogy nem, akkor csak találgatunk..

Szteganalízis

- Általános szteganalízis
  - Tetszőleges algoritmusra működik
  - Kevésbé jó eredmények
- Algoritmus specifikus szteganalízis
  - Csak az adott algoritmusra működik
  - Jó eredmények
- Sikereses, ha felismeri az üzenet jelenlétét
  - Sokszor az üzenet hossza is felismerhető

LSB rejtés felismerése – Andreas Westfeld

- PoV: Pair of Values (Westfeld)
  - Értékpárok LSB mentén (pl. a 2 és a 3)
  - A rejtés hatására (felülírt LSB) az értékpárok előfordulása egyenletes lesz
  - Jó módszer, ha a rejtett üzenet elég hosszú
  - Rövid üzenet esetében csak akkor működik, ha ismert a rejtés helye

LSB rejtés felismerése – Sorina Dumitrescu

- Kép tartományok – Dumitrescu
  - A kép egymás melletti pixeleit párokba rendezzük: P
  - 3 halmazt definiálunk P-n:
    - X: u,v pár esetén vagy *u páros és u>v* vagy *u páratlan és u<v*
      - *u,v: 2k, 2k-m* vagy *2k+1, 2k+1+m*
    - Y: u,v pár esetén vagy *u páros és u<v* vagy *u páratlan és u>v*
      - *u,v: 2k, 2k+m* vagy *2k+1, 2k+1-m*
    - Z: u,v pár esetén u=v
  - Az Y halmazt tovább bontjuk:
    - W: u,v pár esetén (2k,2k+1) vagy (2k+1,2k) alak (távolság=1)
    - V: Y-W

LSB rejtés felismerése – Dumitrescu 2.

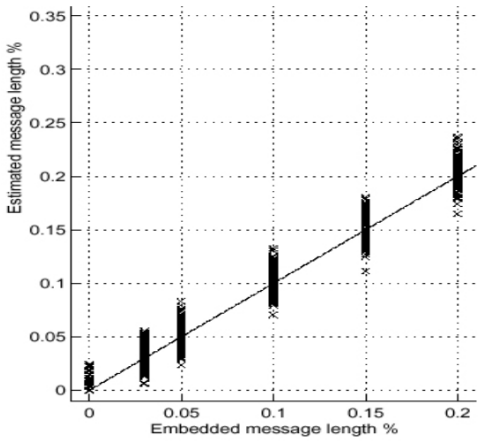
- Ha a kép forrása természetes, akkor statisztikailag | X | = | Y | (megfigyelés)
- Az LSB rejtés megváltoztatja a képpontok helyét a halmazokban. u,v pár esetén:
  - 0,0: u és v nem változnak
  - 0,1: v változik
  - 1,0: u változik
  - 1,1: u és v is változik

LSB rejtés felismerése – Dumitrescu 3.

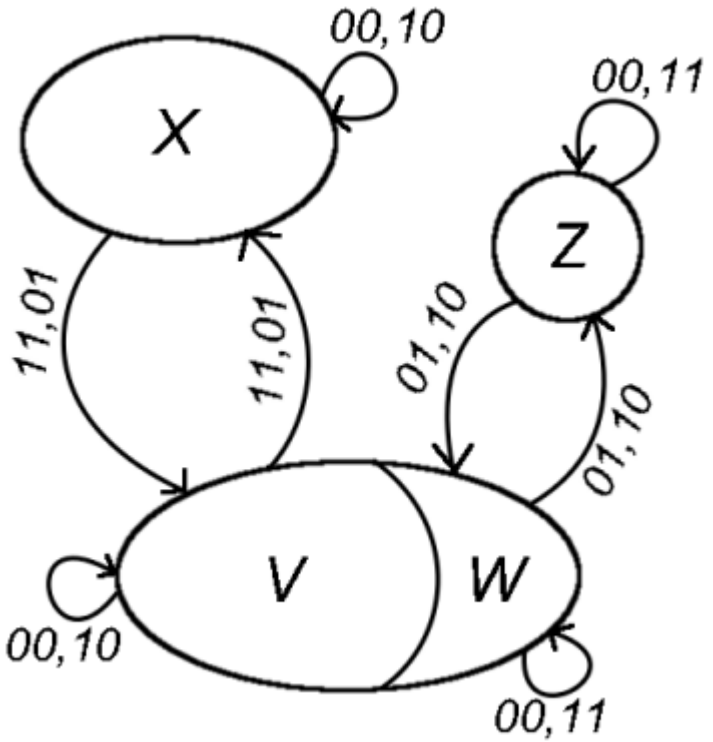
- Változások hatása a halmazokra:
- p bit változására

$0.5\gamma p^2+(2|X'|-|P|)p+|Y'|-|X'|=0$

$\gamma=|W|+|Z|=|W'|+|Z'|$



Eredmény csak, a levezetés nincs itt



Szomszédos színek módszere

- Friedrich:
  - Közeli szín, ha RGB értékben csak 1 LSB bit eltérés van.
  - A közeli színek és egyedi színek aránya jelentősen nő, ha cover image-be rejtünk a stego-image-be rejtéshez képest
    - Cover és stego image megkülönböztethető
- Westfeld:
  - LSB helyett lehet +/-1 is az elrejtés módszere. Erre az LSB módszerek nem működnek.
  - Egy adott színnek 26 szomszédja lehet (mindegyik komponens +/-1). De egy természetes képen a színeknek csak 4-5 szomszédjuk van.