Robusztus adatrejtés

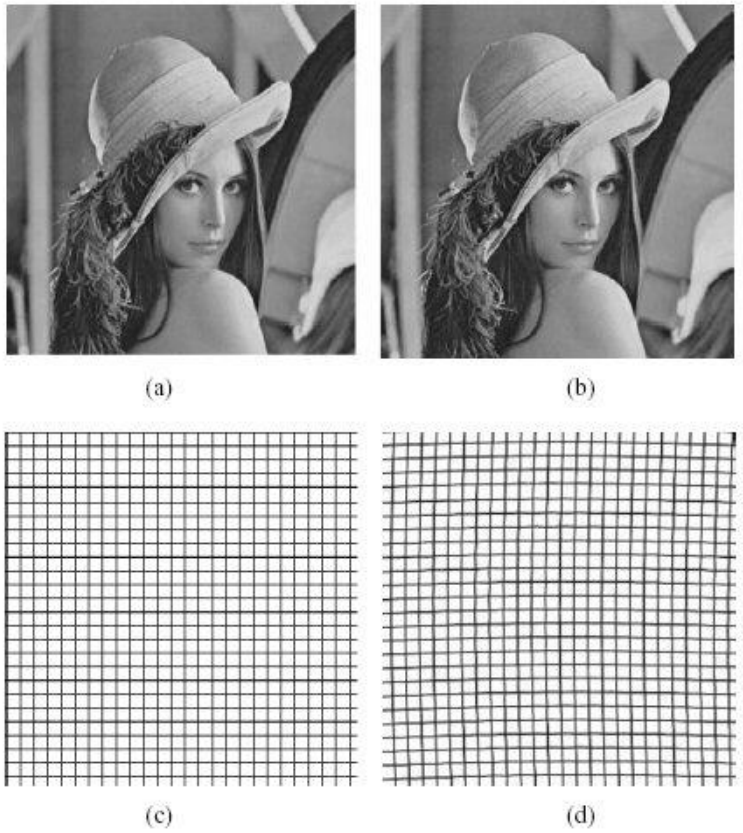
BME - TMIT
VITMA378 - Médiabiztonság
feher.gabor@tmit.bme.hu

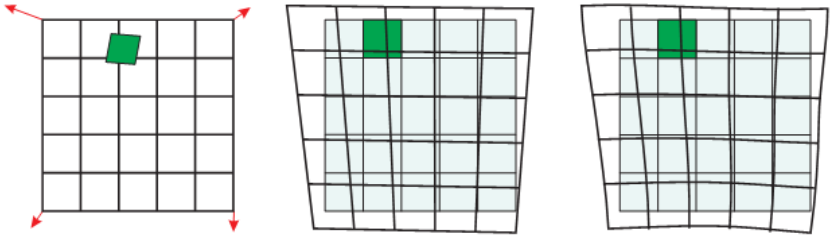
Adatrejtés támadásai

- Adatrejtés eltűntetése a média torzításával
  - D/A és A/D átalakítás, lineáris és nem lineáris transzformációk
  - Vágás, tükrözés, forgatás, méretezés, sortörlés, színtér csökkentése, élesítés, zajszűrés, tömörítés, átkódolás, …
- Automatikus adatrejtés benchmark
  - Azt vizsgáljuk, hogy a torzítást milyen mértékben éli túl a vízjel
    - StirMark, CheckMark, Optimark

Adatrejtés vizsgálat példák

- Stirmark támadás példák





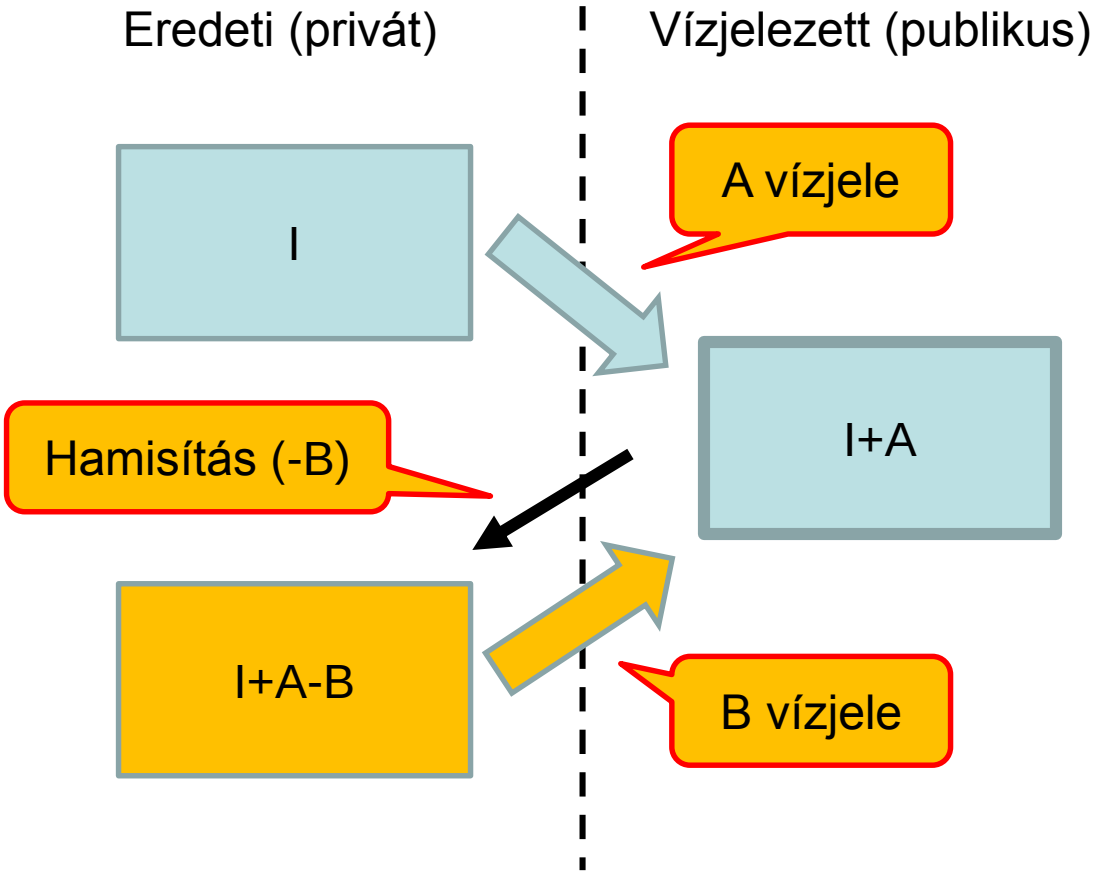
Adatrejtés támadásai 2.

- Kriptografikus támadások
  - Ismerjük az algoritmust, tudjuk, hogy információ van rejtve
  - A stego-kulcs brute-force támadása
- A protokoll támadása
  - Több különböző adatot rejtő hasonló tartalmú média ütköztetése. A vevők összejátszanak
  - Vízjel eltűntetése
    - Pl.: több különbözően vízjelzett ugyanazon média átlagolása
  - Másolásos támadás
    - Egy média vízjelét átmásolják egy másik médiára

Protokoll támadásai – IBM támadás

- IBM támadás
  - Új vízjel helyett egy hamis eredeti előállítása, amelyhez képest a vízjelezett példány a hamis vízjelet is tartalmazza
  - I, az eredeti média, I+A a publikus, vízjelezett média
  - A támadó készít egy hamis eredeti médiát a publikusból: I’ = I+A-B
  - A publikus média így a saját eredeti médiájának a vízjelezett változata (B vízjelet tartalmazza)
    - I+A = I’+B és I = I’+B-A
  - Nem lehet eldönteni, hogy kié az eredeti!

Protokoll támadása



Robusztus video vízjelezés

Nagy torzítások esetén is olvasható maradjon az elrejtett adat

Zhao-Koch algoritmus (1995)

- Jian Zhao és Eckhard Koch
- Adatrejtés a frekvencia tartományban
  - Randomly Sequenced Pulse Position Modulated Code (RSPPMC)
    - A tipikus képek még zaj hatására élvezhetőek maradnak
    - (Digitális) Előállításuknál sok véletlen tényező közrejátszik, nem lehet őket másodszor ugyanúgy előállítani
  - Kiválasztunk két DCT értéket $Y(k_1,l_1)$ és $Y(k_2, l_2)$
    - Vizsgáljuk a kettőjük abszolút értékének különbségét: $|Y(k_1,l_1)| - |Y(k_2,l_2)|$
      - Várható értékben közel 0-t kell, hogy kapjunk
    - „Highs”: $|Y(k_1,l_1)| > |Y(k_2,l_2)| + p$
    - „Lows ”: $|Y(k_2,l_2)| > |Y(k_1,l_1)| + p$
    - Ahol p a JPEG kvantálás és a zaj miatt van (robusztusabb így)

Ilyen egy „normális” kép

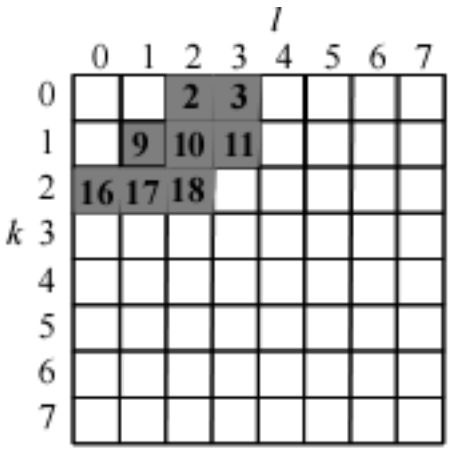
Zhao-Koch algoritmus

- Az adatrejtés során:
  - Álvéletlen alapján kiválasztunk DCT komponens párokat
  - Az elrejtendő információ szerint rendezzük át őket: „highs” és „lows” formára
- Az adat felfedése esetén:
  - Ugyanazt az álvéletlent használva kiválasztjuk a DCT komponenseket
  - Megvizsgáljuk viszonyukat
  - Következtetünk az elrejtett információra

- Előnye, hogy tervezetten (p) jól tűri a további tömörítést és a képfeldolgozást, ahol szűrést használunk

Zhao-Koch továbbfejlesztés

- Kettő helyett több Y(k,l) értéket is figyelembe lehet venni.
  - Több felosztás: „high”, „low” és „middle”
  - Nem valós állapot is (ha túl sok változtatás szükséges)
- Adatrejtés közép frekvenciába
  - A nagy frekvenciában kevésbé ellenálló, a kis frekvenciában pedig észrevehető



(k_1,l_1)	(k_2,l_2)	(k_3,l_3)	
H	M	L	patterns for ’1’
M	H	L	
H	H	L	
M	L	H	patterns for ’0’
L	M	H	
L	L	H	
H	L	M	invalid patterns
L	H	M	
M	M	M	

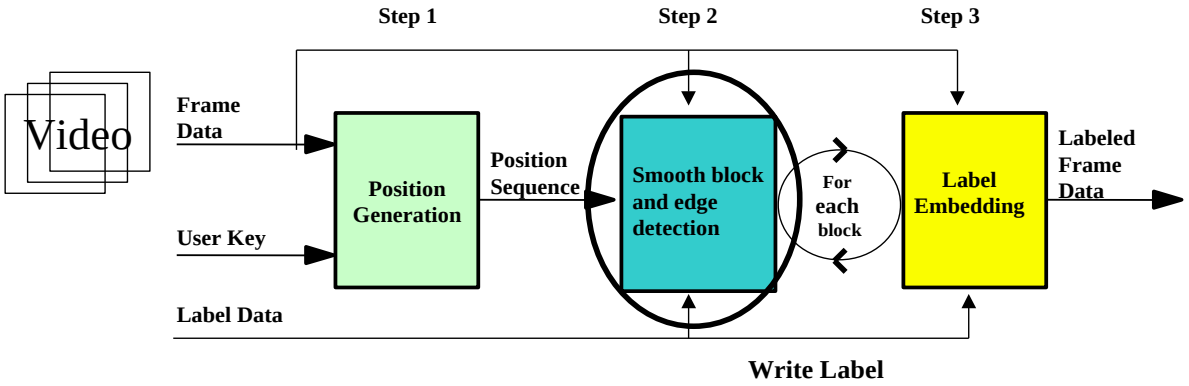
Set No.	(k_1,l_1)	(k_2,l_2)	(k_3,l_3)
1	2(0,2)	9(1,1)	10(1,2)
2	9(1,1)	2(0,2)	10(1,2)
3	3(0,3)	10(1,2)	11(1,3)
4	10(1,2)	3(0,3)	11(1,3)
5	9(1,1)	2(0,2)	10(1,2)
6	2(0,2)	9(1,1)	10(1,2)
7	9(1,1)	16(2,0)	2(0,2)
8	16(2,0)	9(1,1)	2(0,2)
9	2(0,2)	9(1,1)	16(2,0)
10	9(1,1)	2(0,2)	16(2,0)
11	10(1,2)	17(2,1)	3(0,3)
12	17(2,1)	10(1,2)	3(0,3)
13	10(1,2)	3(0,3)	17(2,1)
14	3(0,3)	10(1,2)	17(2,1)
15	9(1,1)	16(2,0)	17(2,1)
16	16(2,0)	9(1,1)	17(2,1)
17	10(1,2)	17(2,1)	18(2,2)
18	17(2,1)	10(1,2)	18(2,2)

Dittmann-I algoritmus (1998)

- Jana Dittmann
- Adatrejtés DCT tartományban (Zhao-Koch továbbfejlesztése)

- Adaptív adatrejtés
  - Célunk olyan DCT blokkba mentés, ahol sok a frekvencia komponens, így az adatrejtés torzításai jobban tolerálhatóak
  - Az éleknél is sok a frekvencia komponens, de itt szembetűnőbb a torzítás

- A blokk „simaságát” (smooth) és az éleket nézzük
  - Gyors megoldás kell, mert valós időben szeretnénk dolgozni



Dittmann-I algoritmus 2.

- Smooth (sima):
  - A nem 0 értékű DCT kitevők száma egy újabb kvantálás után (Q_m)
- Élek
  - Az alacsony frekvenciájú együtthatók abszolút értékének összege (1,2,8,9,10,16,17)

low	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	High

Q_m

Dittmann-I algoritmus 3.

- Az adatrejtés (torzítás) iránti tolerancia:
  - level = smoothscale * smooth + edgescale * edge + offset
    - offset: az adatrejtés alap erőssége
    - Példa étékek:

Marad alacsony frekvencia, ez jó

smoothscale = -10
edgescale = 0.27
offset = 50

Sok az él, nem olyan jó

  - A „level” értékét 0 és 50 közé korlátozzuk
  - Minél kisebb az érték, annál nagyobb módosítás lehetséges

- Az adatrejtés erősségét Q_f már a DCT blokkra jellemző „level” értékhez igazítjuk
  - DCT érték módosítás Q_m/Q_f paraméterrel

Level	0-9	10-19	20-29	30-39	40-49
Q_f	1	1	2	3	4

Dittmann-I algoritmus 4.

- Az adatrejtés előtt hibajavító kódolás és redundancia bevitele ismétléssel
  - (31, 6, 15)-BCH kódolás (Bose, Ray-Chaudhuri, Hocquenghem)
  - Ismétlés 31 bitenként
- Adatrejtés DCT blokkba:
  - Egy bit elrejtésére 3 középfrekvenciás értéket választunk
    - Y1, Y2 és Y3 abszolút-értékkel
    - A táblázat szerint megváltoztatjuk az értékeket
    - Ha túl nagy a változás, akkor egy a táblázatban nem szereplő éréket választunk
    - Az érvényes kombinációk közül a legkisebb változást vesszük

Bit	1	1	1	1	0	0	0	0
Y1	H	H	M	M	L	L	M	M
Y2	H	M	H	M	L	M	L	M
Y3	L	L	L	L	H	H	H	H

Dittmann-I algoritmus 5.

- Tesztek eredményei:
  - Az adatrejtés megfelelően rejtett
  - További tömörítés esetén:
    - MPEG2: 1-5% hiba
    - Quicktime: 1-7% hiba
    - StirMark: 32% hiba

Patchwork

- Egyetlen bit elrejtése egy képben
  - Két foltot határozunk meg a képen
    - Az egyik folt fényességét növeljük
    - A másik folt fényességét csökkentjük
  - Egy kép esetén sok pont fényességének a különbsége közel 0
    - $\sum_n(a_i-b_i) \approx 0$ nagy n esetén
  - Ha megváltoztatjuk *n* pont fényességet párban $\pm\delta$ -val, akkor ez a különbség $2n\delta$ lesz ezt az *n* pontot vizsgálva
  - Az egyetlen bitnyi információ a vízjel jelenléte

Fridrich algorithm (1997)

- Jessica Fridrich
- „Overlay pattern”
- Egy egész kép terjedelmű minta előállítása
  - A minta álvéletlen alapján áll elő
  - Nem tartalmaz magas frekvenciákat (szűrés)
  - Nagyobb 1 és 0 blokkok a mintában
- A minta skálázása és összeadása az eredeti képpel

- A detektáláshoz szükséges az eredeti kép
  - Ellenőrizzük, hogy megtalálható benne a minta

- Jól tűri a torzításokat

Dittmann-II algoritmus (1998)

- Fridrich algoritmus továbbfejlesztése
- Adatrejtés a képtartományban (spatial)
  - Nem az egész képet használjuk, hanem blokkokat
  - 8x8 blokkot használunk, de nem DCT!
- Adatot és nem csak mintát rejtünk el
- A kiolvasáshoz nem szükséges az eredeti média
  - Valós idejű videó esetében nem áll rendelkezésre az eredeti média

Dittmann-II algoritmus

- Az algoritmus lépései:
  - Blokkok kiválasztása álvéletlen alapján
  - Álvéletlen 8x8 blokk előállítása magas frekvenciák nélkül
    - Nincs gyors változás, nem zajszerű
  - Az elrejtendő adat redundanciája
    - Hibajavító kódolás
    - Ismétlés
  - Adatrejtés a luminancia értékek közé. Az adatrejtés erőssége a „level” érték alapján
    - „smooth” és él számítás

Dittmann-II algoritmus 2.

- Álvéletlen 8x8 M-minta előállítása
  - A blokkba 0 és 1 értékek álvéletlen alapján (stego-kulcs)
  - A magas frekvenciák eltávolítása:
    - Ha egy adott pont szomszédjai közül több mint 5 „1” érték, akkor 1 marad/lesz
    - Ha egy adott pont szomszédjai közül kevesebb mint 3 „1” érték, akkor 0 marad/lesz
    - Ismétlés, amíg van változás
  - A blokk egyértelműen előáll

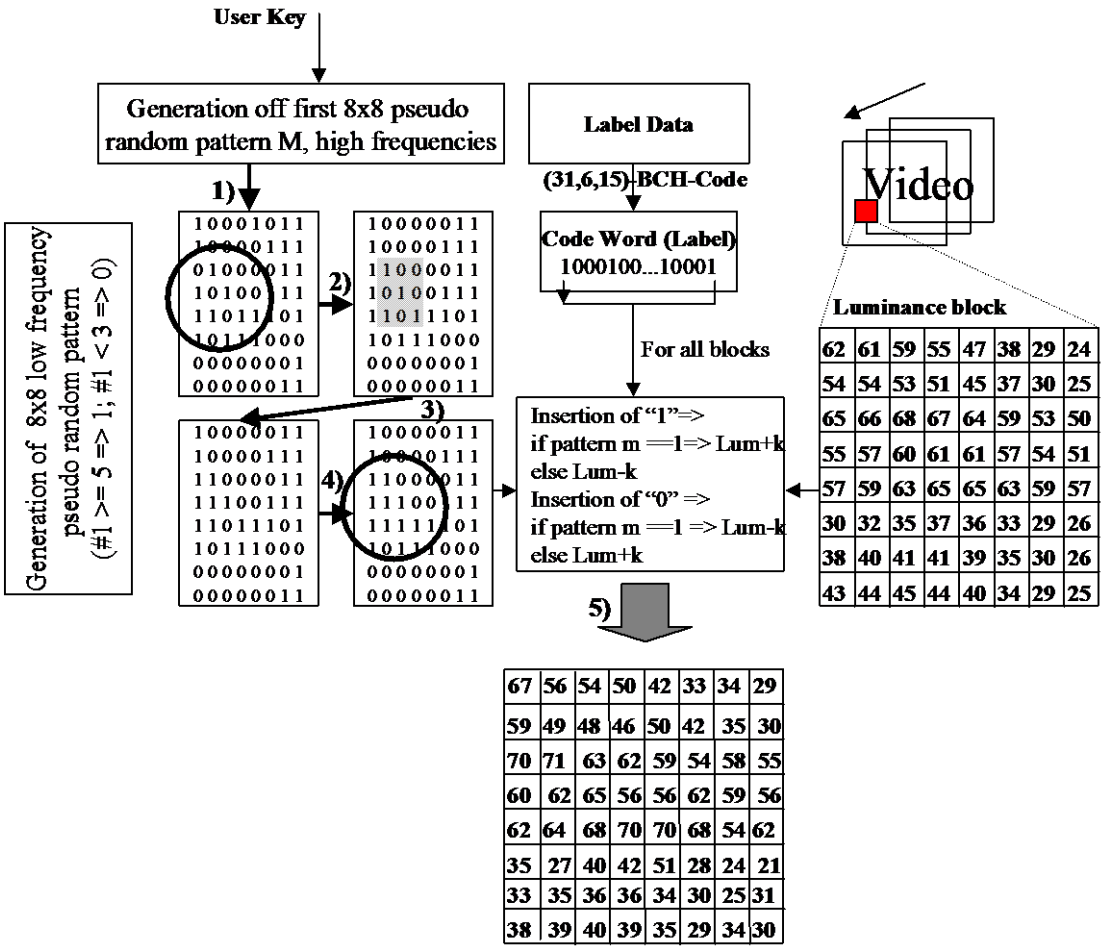
Dittmann-II algoritmus 3.

- Adatrejtés
  - „k” érték meghatározása a blokk „smooth” és él tulajdonságai alapján (Q_f mintájára)

Level	0-9	10-19	20-29	30-39	40-49
k	12	12	6	4	3

  - Korreláció előidézése
    - Ha a rejtendő bit „1”:
      - Mindenhol ahol a 8x8 M-minta 1, ott a luminancia értékek k –val növelése, a többi érték k –val csökkentése
    - Ha a rejtendő bit „0”:
      - Mindenhol, ahol a 8x8 M-minta 1, ott a luminancia értékek k –val csökkentése, a többi érték k –val növelése
  - 1 bit elrejtése 1 db 8x8 blokkban

Dittmann-II példa



Dittmann-II algoritmus 4.

- Rejtett adatok felfedése:
  - Blokkok kiválasztása álvéletlen alapján
  - M-minták előállítása
  - Korreláció megállapítása
  - Ha van korreláció, akkor a rejtett adat bitjének meghatározása
  - A rejtett adat meghatározása

Dittmann-II algoritmus 5.

- Korreláció megállapítása
  - Az 8x8 M-minta alapján
    - av1: A 8x8 luminancia blokk értékeinek átlaga, ahol az M-minta 1 értékű
    - av0: A 8x8 luminancia blokk értékeinek átlaga, ahol az M-minta 0 értékű
  - Ha korrelálatlan, akkor av1 – av0 értéke 0 körül van
  - Ha korrelált, akkor a differenciájuk 2k körül van
  - A rejtett információ:
    - „1” bit, ha av1 > av0
    - „0” bit, ha av1 < av0

Dittmann-II algoritmus 6.

- Tesztek eredményei:
  - Az adatrejtés megfelelően rejtett
  - További tömörítés esetén:
    - MPEG: 1-7% hiba
    - Quicktime: 8-35% hiba
    - StriMark: 31% hiba
  - A geometriai torzításokat jobban tudjuk kezelni
    - Az M-mintát is torzítani kell

Transzformáció más tartományba

- Nem képi tartományban történik az adatrejtés
  - DCT tartomány
    - Ha alacsony komponenseket használok, akkor jobban ellenáll az átkódolásnak
- RST (Rotation - Scale – Translation) invariáns adatrejtés
  - FMT – Fourier – Mellin transzformáció
    - A Fourier transzformáció ellenáll az eltolásnak
    - Mellin transzformációnál a log-polar koordinátáknál a forgatás és átméretezés eltolásnak hat
      - Az eredeti kép polár koordinátái log(r) és φ esetén a transzformáció koordinátái: r és φ
    - Egy végső Fourier transzformáció így eltűnteti a forgatás és átméretezés hatását
    - Az FMT transzformáció ezért robusztus a forgatással, átméretezéssel és eltolással szemben

DFT / FFT

- 2D diszkrét Fourier transzformáció

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(\frac{ux}{M}+\frac{vy}{N})}$$

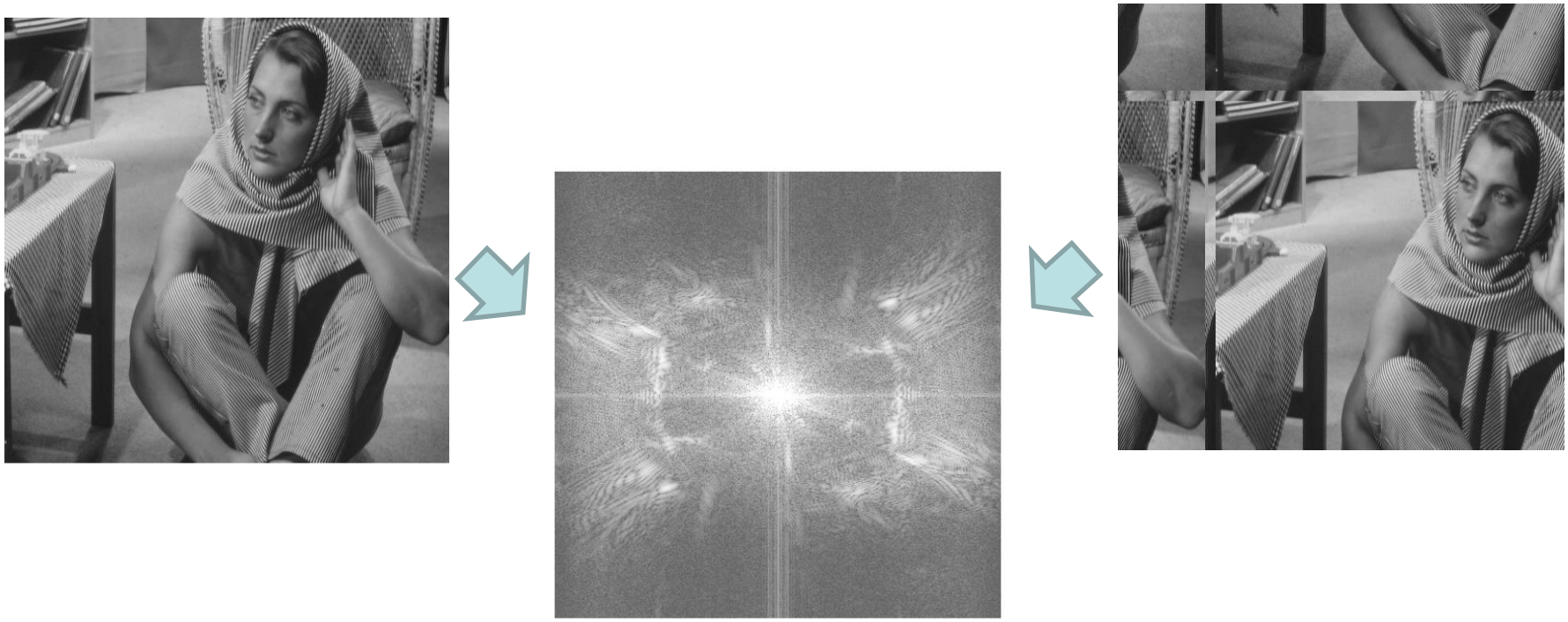
$$(u = 0, 1, \ldots, M-1, v = 0, 1, \ldots, N-1)$$

$$f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi(\frac{ux}{M}+\frac{vy}{N})}$$

$$(x = 0, 1, \ldots, M-1, y = 0, 1, \ldots, N-1)$$

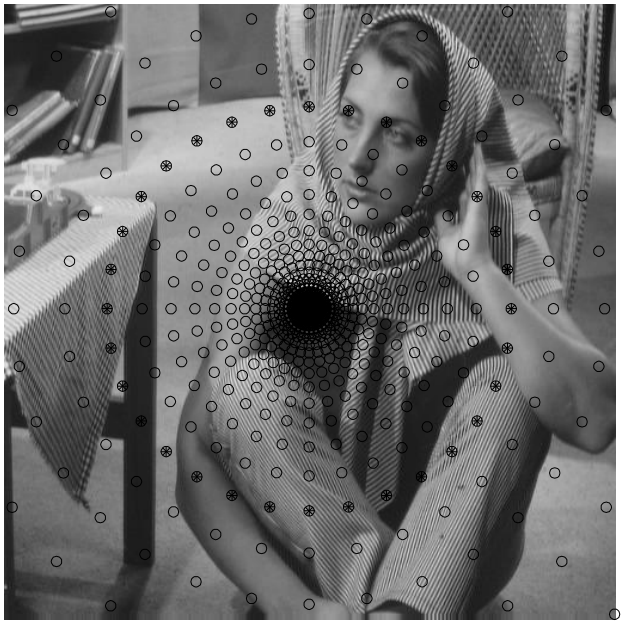
FM transzformáció

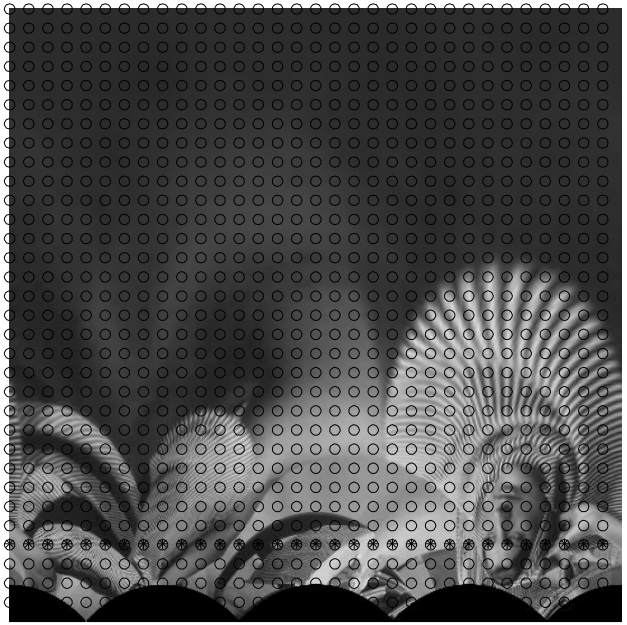
- Fourier transzformáció
  - Eltolás invariancia (Spektrumra)



Log-polar Mapping

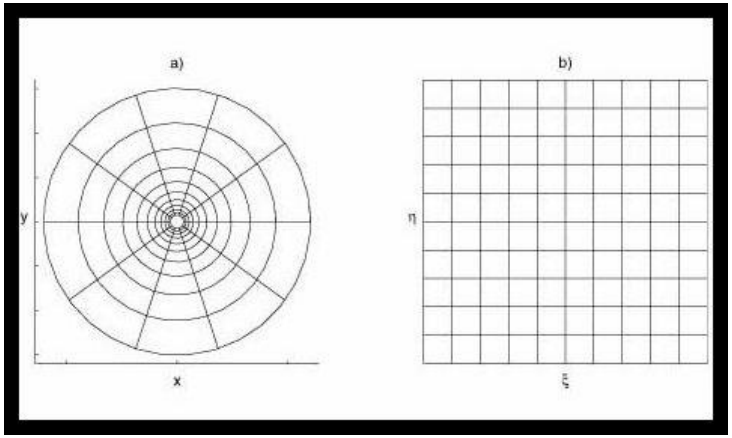
- $x = e^r \cos(\varphi)$, $y = e^r \sin(\varphi)$
- LPM + Fourier -> Fourier Mellin





Log-polar Mapping 2.

- Középen az origó
  - $x = e^r \cos(\varphi)$, $y=e^r \sin(\varphi)$
  - $r=\ln\sqrt{(x^2+y^2)}$, $\varphi=\operatorname{atan}(y/x)$



Log-polar Mapping 3.

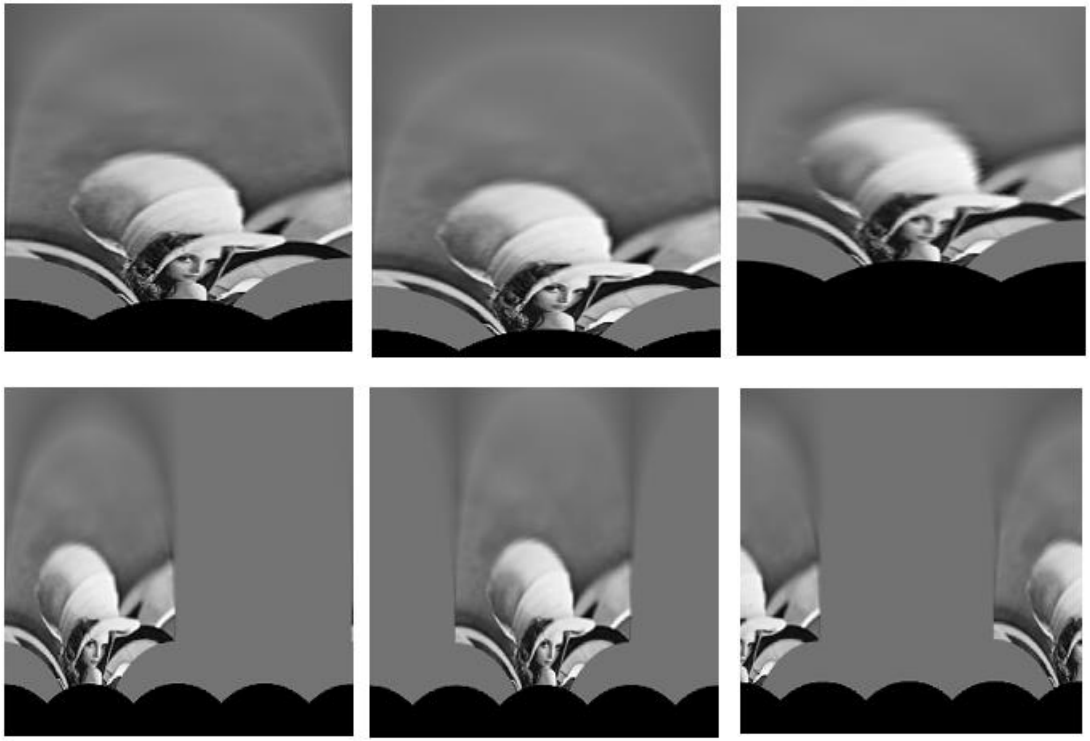
- Log-polar mapping képeknél
  - x, y, x_c, y_c: Descartes koordináták és közepek
  - R, W, n_r, n_w : Gyűrű és szektor koordináták és számok (ennyi darab gyűrű/szektor)
  - r_{min}, r_{max}: A legkisebb és legnagyobb gyűrű mérete a Descartes rendszerben

  - $r = \sqrt{((x-x_c)^2 + (y-y_c)^2)}$, $\varphi = \text{atan}((y-y_c) / (x-x_c))$
  - $R = (n_r - 1)\ln(r/r_{min})/\ln(r_{max}/r_{min})$, $W = n_w \varphi/2\pi$

  - Pontosabb mintavételhez: $r_{min} = r_{max}e^{(-2\pi(n_r - 1)/ n_w)}$

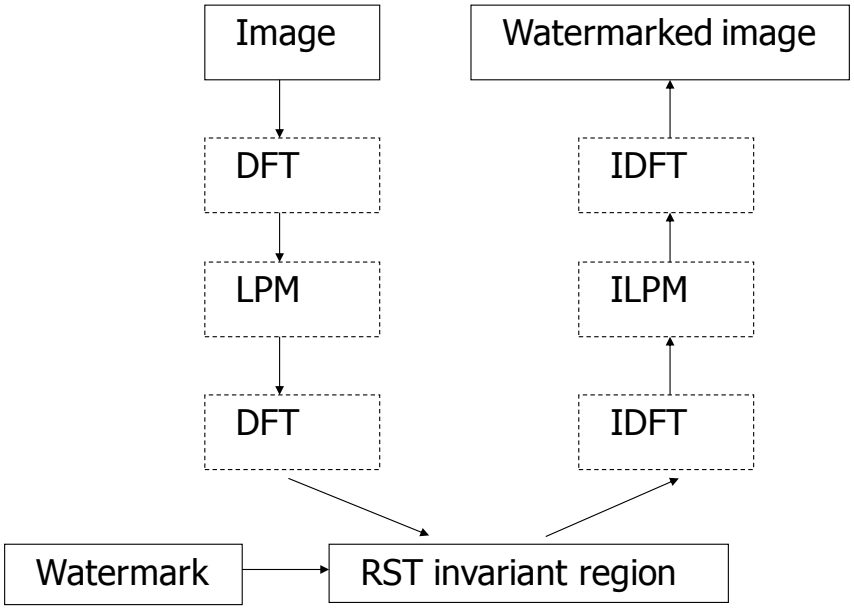
LPM példák

- Lena méretezés és forgatás

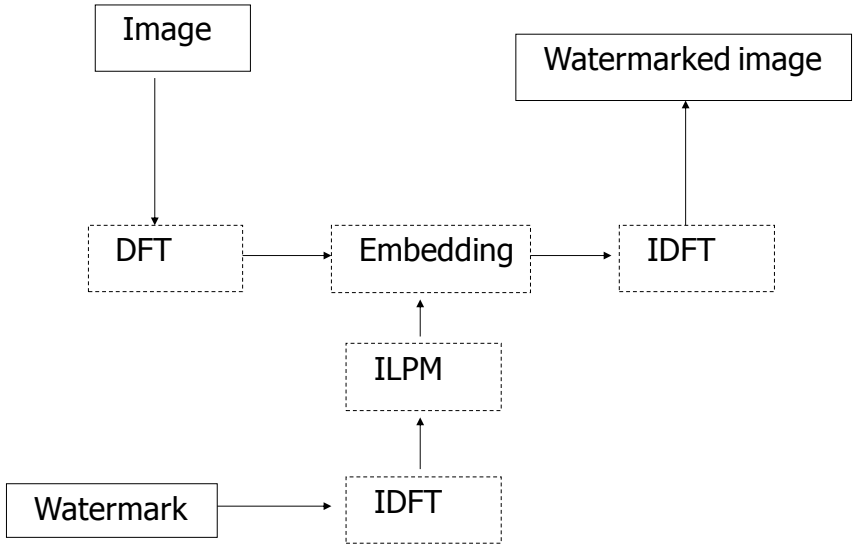


FMT módszerek

1. Módszer



2. Módszer



FMT eredmények

Vízjel szinkronizáció

Szinkronizáció

- A videó esetében a detektálást szinkronizálni kell a médiához
  - TV adás esetén nincs kezdeti szinkronizáció
  - Adatvesztés esetén kiesik a szinkron
  - Ismerni kell az aktuális stego-kulcsot

Szinkronizációs megoldások

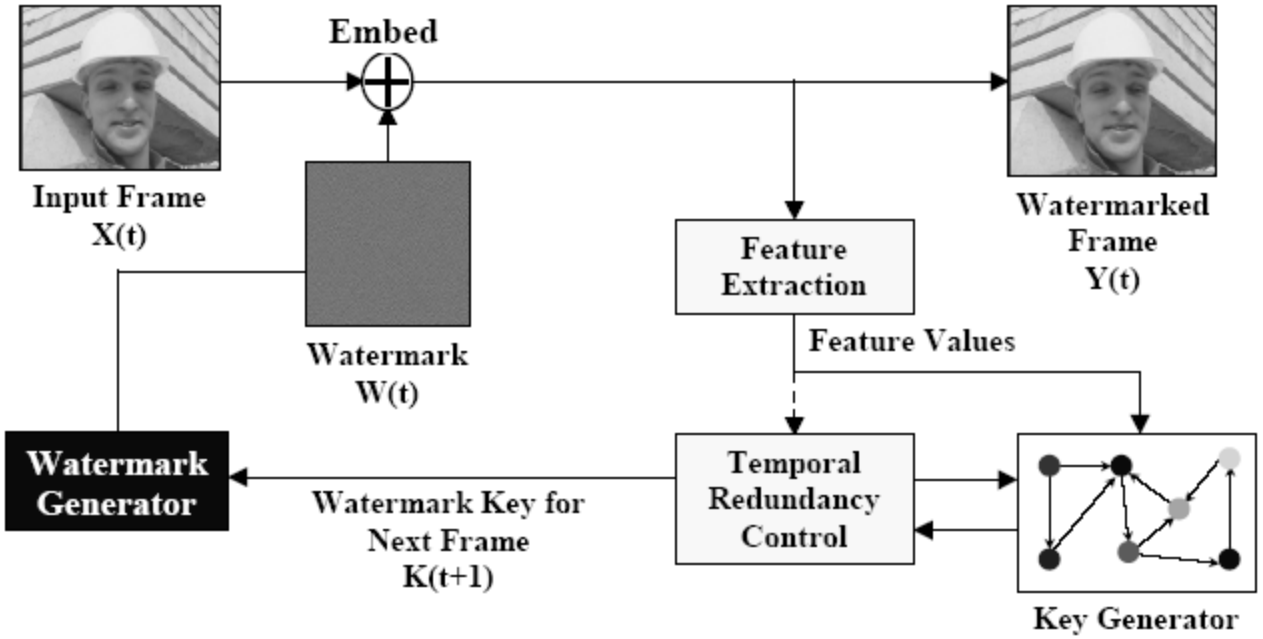
- Időfüggetlen kulcs: Mindig ugyanazt a kulcsot használom minden egyes képkockához
  - Nincs gond a szinkronizációval
  - Alacsony biztonság
    - Támadás: több képkocka átlagolása
- Független kulcsok: Mindig más kulcsot használok minden egyes képkockához
  - Magas biztonság (robusztusság) az elrejtett adat számára
  - A kulcs-szinkront könnyű támadni: képkocka elhagyás vagy csere
  - Az elvesztett szinkront nehéz megtalálni

Szinkronizációs megoldások 2.

- Periodikus kulcsok: Egy alap kulcs időről időre visszatér egyes képkockákon
  - Az adatrejtés számára kevésbé biztonságos, mint a független kulcsok
  - Az elveszett szinkron megtalálható, mert egyszerre a kulcsok egy kis halmazával dolgozunk csak

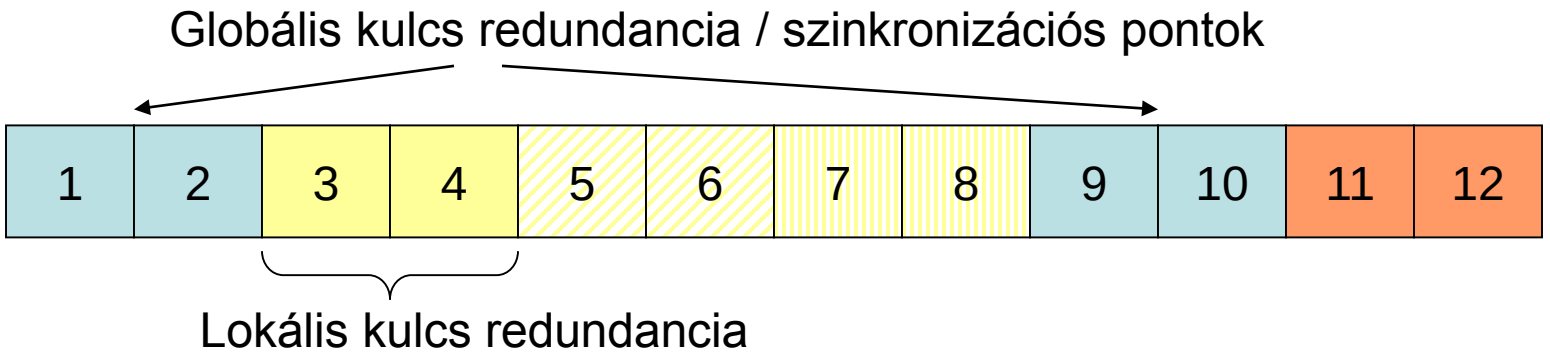
Kulcs szinkronizáció

- Adatrejtés/vízjel beillesztése



Kulcs redundancia

- Időbeli (temporal) redundancia irányítása
  - Néhány egymást követő képkockán megismétlődik az aktuális kulcs (lokális redundancia)
  - Egy hosszabb sorozat után visszatér az alap kulcs (globális redundancia)
  - Az alap kulcsok szinkronizációs pontok, de a szinkronizáció után nem feltétlenül ismétlődik a kulcssorozat



Kulcs redundancia 2.

- A globális redundancia
  - Segít azt elvesztett szinkron megtalálásában
  - Kezdeti szinkronizálás
  - Ha túl nagy, akkor ritkán van lehetőség szinkronizációra
- Lokális redundancia
  - Segít áthidalni támadások/hibák hatását
  - Képkocka eldobásánál nem esik ki a szinkron
  - Ha túl kicsi, akkor szinte független kulcsok (nem jó)

- Optimális értékeket kell találni
  - Adatrejtés biztonság és szinkronizációs biztonság egyensúlya
  - Pl.: 150/5 jónak tűnik némely alkalmazásokban

Videó tulajdonságok a kulcsban

- Feature Extraction
  - Az alapkulcsot követő kulcs függ a videótól
    - A videó bizonyos tulajdonságait visszacsatoljuk a kulcsba
  - A visszacsatolásra használt adatoknak robusztusnak kell lenniük
    - Ne lehessen a képet megtámadni és így elrontani a kulcs-szinkront
    - Valós időben számolható legyen
  - Az előállt kulcsnak titkosnak kell lennie
    - Nem egy az egyben használom a kinyert értékeket
    - Kombinálom más (csak a felek között ismert) kulcsokkal

Videó tulajdonságok a kulcsban

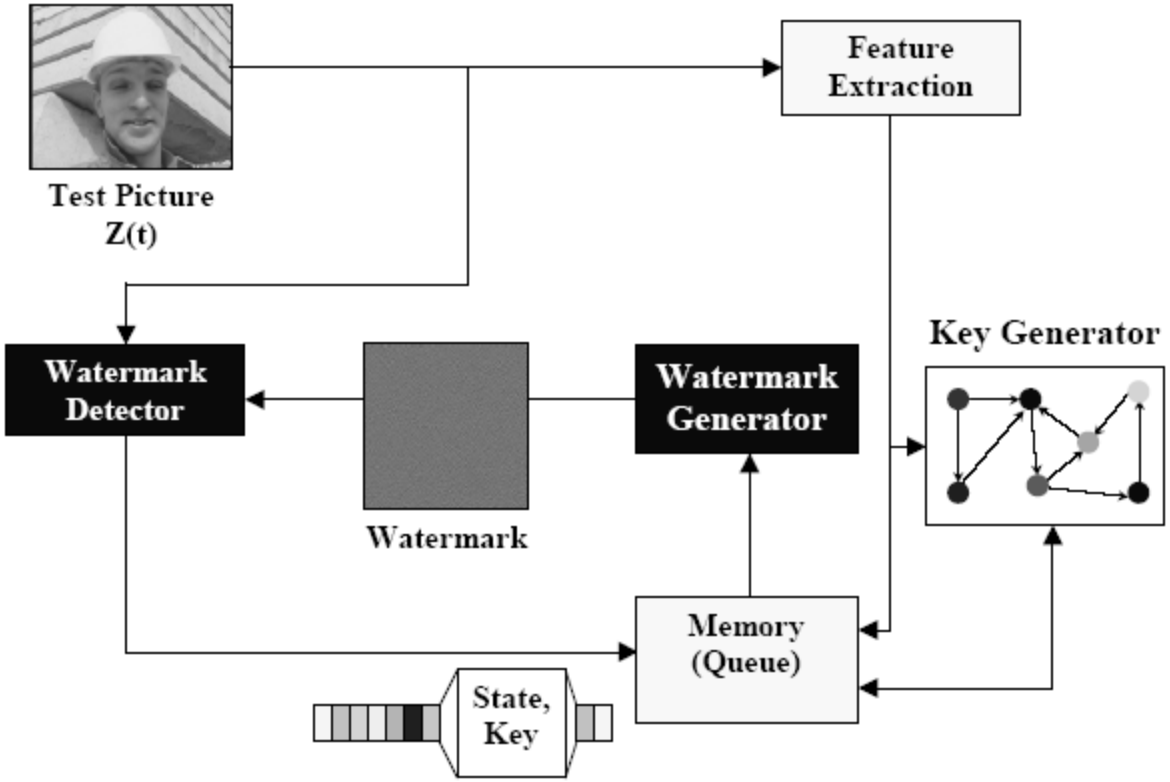
- Példa videó tulajdonság kinyerésére
  - A képteret 32x32 blokkokba osztom
  - Minden blokk véletlenszerűen X, Y vagy Z csoportba tartozik. A felosztás állandó
  - x érték az X csoportba tartozó blokkok fényességének átlaga + kvantálva a robusztusság miatt
  - y és z hasonlóan
  - A videó tulajdonságok az adott képkockára x, y és z értékek

Kulcs generátor

- Kulcs generátor
  - Cél, hogy ne kelljen a kulcsot kimerítően keresni
    - Az egymást követő kulcsok megjósolhatóak a már használt kulcsokból
    - Az aktuális kulcs keresésékor limitált kulcsot kell végignézni, nincs „keresés”
  - A videó bizonyos tulajdonságai itt is használhatóak (Feature Extraction)
  - Kulcsgenerátorok:
    - Állapotgépek
    - Egyirányú függvények

Kulcs szinkronizáció

- Adatrejtés/vízjel detektálása



Kulcs memória

- Memory (Queue)
  - Ha jön egy új képkoca, akkor az alapkulccsal vagy a memóriában tárolt kulcsokkal próbálkozunk
  - Ha nincs, akkor megpróbáljuk a kulcsot kitalálni, jövőbeli kulcsok gyártásával
  - Ha ez sem sikerül, akkor jöhet a következő képkocka és újra próbálkozunk