Prelucrarea Imaginilor

Curs 12

Compresia Imaginilor

19 Decembrie 2019

Compresia imaginilor

Compresia imaginilor este necesara la reprezentarea in format numeric (in scopul transmiterii sau stocarii) deoarece consuma o cantitate mare de informatie. Termenul de compresie se refera la totalitatea metodelor ce au drept scop reducerea cantitatii de date necesare pentru reprezentarea unei imagini. Compresia este folosita in special pentru stocarea sau transmiterea imaginilor. De exemplu cazul unei imagini cu o rezolutie 640x480 pixeli, imaginea contine 307200 puncte, fiecare punct necesitind 24 biti (pentru o reprezentare de calitate a culorii). Rezulta o dimensiune de 921.600 octeti (900KB) ceea ce este destul de mult (Reprezentarea numerica a imaginilor).

Pentru a stoca o imagine avem nevoie de spatiu considerabil, iar pentru transmiterea ei avem nevoie de un canal de transmisiune de banda larga, de care nu dispunem intotdeauna.

Cel mai frecvent imaginile se codifica intr-o forma comprimata existind in momentul de fata un numar mare de tehnici de compresie si un numar si mai mare de formate (GIF, JPEG, WIF...).

Clasificarea metodelor de compresie

Metodele de compresie pot fi: http://www.miv.ro/ro/documentatie/pi/PIlab09.pdf

1. Metode de compresie la nivel de pixel

Aceste metode nu tin cont de corelatia care exista intre pixelii vecini, codand fiecare pixel ca atare. Acest tip de compresie este fara pierdere de informatie, adica imaginea initiala poate fi refacuta perfect din imaginea comprimata. Astfel sunt: codarea Huffman, LZW (Lempel-Ziv-Walsh) si RLE (Run Length Encoding)

2. Metode de compresie predictive

Acestea realizeaza compresia folosind corelatia dintre pixelii vecini: codarea cu modulatie ***delta*** si DPCM (Differential Pulse Code Modulation).

3. Metode de compresie cu transformate

Aceste metode se bazeaza pe scrierea imaginii intr-o alta baza, prin aplicarea unei transformari unitare, atfel incat energia imaginii sa fie concentrata intr-un numar cat mai mic de coeficienti.

4. Alte metode sunt:

cuantizarea vectoriala, codarea folosind fractali, codarea hibrida.

Reprezentarea numerica a imaginilor

http://www.indinf.pub.ro/catalinp/img/bmp.htm

O imagine este o suprafata de obicei dreptunghiulara caracterizata, la nivelul oricarui punct al ei, de o anumita culoare. La modul ideal, culoarea variaza in mod continuu in oricare directie. Din pacate, in sistemele numerice, nu se pot utiliza marimi care variaza continuu ci doar forma discretizata a acestora.

Discretizarea este operatia prin care se reprezinta o marime cu variatie continua sub forma unui ansamblu finit de esantioane.

De exemplu, temperatura mediului ambiant variaza in mod continuu pe durata unei zile, totusi se poate reprezenta evolutia acestiea prin intermediul valorilor masurate la un interval de un minut.

Se poate observa ca pentru a se obtine o captare cit mai fidela a evolutiei temperaturii aceasta trebuie masurata la un interval cit mai scurt posibil. In caz contrar (spre exemplu daca s-ar masura o data pe ora) se pot pierde anumite variatii care au aparut intre doua masuratori si care astfel nu mai pot fi observate.

… Reprezentarea numerica a imaginilor

In concluzie, o imagine trebuie sa fie discretizata inainte de a se pune problema reprezentarii numerice.

Discretizarea consta in impartirea imaginii intr-un caroiaj asemanator unei table de sah. Fiecare sectiune de imagine delimitata de acest caroiaj va fi considerata ca avind o culoare uniforma - o medie a culorii existente pe aceasta sectiune. Aceste sectiuni mai sint denumite si pixeli sau ppuncte, numarul acestora definind rezolutia imaginii.

Astfel se poate afirma, de exemplu, ca o imagine oarecare are o rezolutie de 640x480 pixeli ceea ce inseamna ca pe suprafata acesteia s-a definit un caroiaj care o imparte pe orizontala in 640 de sectiuni iar pe verticala in 480.

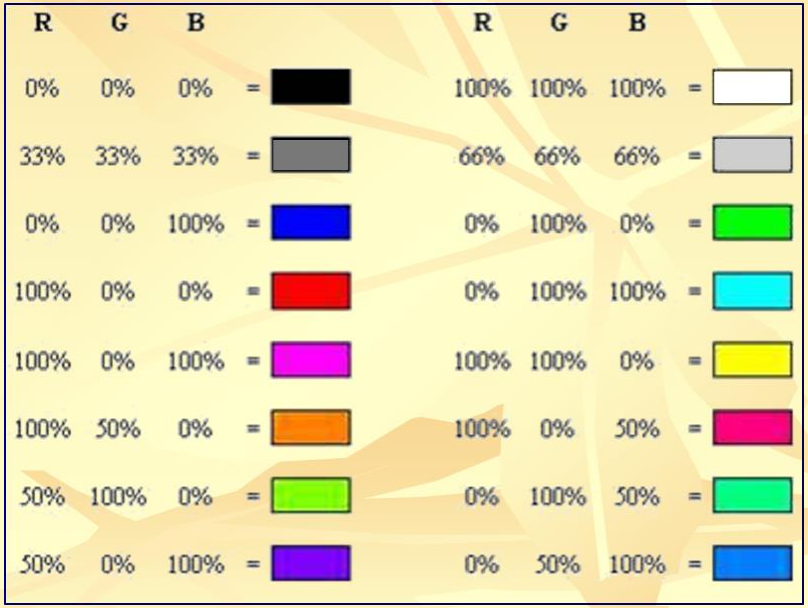
Operatia de discretizare a unei imagini este pusa in evidenta in imaginea de mai jos care a fost marita in mod special pentru a se putea observa caroiajul.

… Reprezentarea numerica a imaginilor

Pasul urmator il constituie gasirea unei reprezentari pentru culoare. Orice culoare poate fi descompusa in trei culori primare (de exemplu rosu-R, verde-G si albastru-B), cu alte cuvinte orice imagine poate fi obtinuta prin suprapunerea aditiva a trei radiatii luminoase avind aceste trei culori si intensitati diferite.

In figura urmatoare se prezinta obtinerea a citeva culori prin acest mecanism:

Deci, pentru a reprezenta numeric o culoare este suficient sa reprezentam intensitatile luminoase ale celor trei culori primare. Daca alocam cite 8 biti pentru fiecare componenta se pot codifica 256 nivele de intensitate, astfel, absenta culorii (intensitate zero) se codifica prin valoarea 00000000 binar (00h) iar intensitatea maxima prin cea mai mare valoare reprezentabila pe 8 biti si anume 11111111 binar (FFh).



… Reprezentarea numerica a imaginilor

Aceasta reprezentare, insa, tine mai mult de modalitatile tehnice de captare si reproducere a imaginii si mai putin de mecanismul fiziologic de percepere a culorii.

Prin diferite experimente s-a constatat din punct de vedere al capacitatii de percepere a detaliilor, ochiul este mai sensibil la intensitatea luminoasa a culorii decit la nuanta.

Din acest motiv este interesanta o alta modalitate de reprezentare a culorii care sa tina cont de aceasta observatie, un exemplu fiind reprezentarea YUV utilizata in televiziunea in culori. In acest caz, in locul celor trei componente primare R,G,B se utilizeaza alte trei marimi derivate din acestea si anume:

$$Y = 0.30*R + 0.59*G + 0.11*B$$

$$U = R - Y = 0.70*R - 0.59*G - 0.11*B$$

$$V = B - Y = -0.30*R - 0.59*G + 0.89*B$$

… Reprezentarea numerica a imaginilor

In cazul acestei reprezentari, componenta Y corespunde intensitatii luminoase percepute pentru respectiva culoare (coeficientii 0.30, 0.59 si 0.11 sint stralucirile relative la alb ale celor trei culori primare rosu, verde respectiv albastru). Aceasta componenta mai este intilnita si sub numele de luminanta.

Componentele U si V sint cele care definesc nuanta culorii, din acest motiv, sint denumite componente de crominanta. Se observa ca se calculeaza ca diferenta dintre componenta rosie respectiv albastra si cea de luminanta.

In cazul televiziunii in culori, acest sistem de reprezentare a fost ales din considerente de compatibilitate cu sistemul de televiziune alb-negru care deja utilizat pe scara larga.

Avantajul reprezentarii YUV este acela ca separa componenta de luminanta pentru care ochiul este foarte sensibil la detalii de componentele de nuanta pentru care sensibilitatea este mai redusa.

Acest lucru face posibila reducerea informatiei asociate unei imagini prin utilizarea unei rezolutii mai reduse pentru componentele de crominanta. In cazul televiziunii in culori se realizeaza o "compresie" prin limitarea benzii de frecventa alocate semnalelor de crominanta (de exemplu in sistemul PAL semnalele U si V au o banda de 1.3Mhz fata de semnalul Y care are o banda de 6Mhz).

Formatul BMP (bitmap)

Acest format este cel mai utilizat la ora actuala pentru reprezentarea numerica a imaginilor. De fapt sub acest nume se regasesc mai multe tipuri de formate care au urmarit de fapt evolutia in domeniul dispozitivelor de afisare a imaginii. In toate cazurile formatul este compus dintr-un antet care contine informatii generale despre imagine (dimensiuni, modul de codificare al culorii, etc.) si o sectiune in care se codifica propriu-zis imaginea. In anumite cazuri, intre aceste doua sectiuni se insereaza o "tabela de culori".

Antetul ocupa 54 de octeti si are urmatoarea structura:

```
typedef struct{ word Type; /* tipul fisierului "BM" */
        dword Size;         /* dimensiunea fisierului (in octeti) */
        word Res1;          /* rezervat (=0) */ word Res2; /* rezervat (=0) */
        dword Offset;       /* adresa relativa in fisier a codificarii imaginii */
        dword Length;       /* numar de octeti pina la sfirsitul antetului (40) */
        dword Width;        /* latimea imaginii in pixeli */
        dword Heigth;       /* inaltimea imaginii in pixeli */
        word Planes;        /* nespecificat (=1) */
        word Color;         /* numar de biti/pixel (culoare) */
        dword Comp;         /* tip compresie (este 0=necomprimat) */
        dword Size;         /* numarul de octeti din codificarea imaginii */
        dword XDef;         /* definitie pe orizontala in pixeli/metru */
        dword YDef;         /* definitie pe verticala in pixeli/metru */
        dword ClrUsed;      /* numarul de culori utilizate in imagine */
        dword ClrImp;       /* numarul de culori importante */
}ANTET;
```

Modul de reprezentare al culorii este indicat de catre cimpul Color si poate fi 1, 4, 8 sau 24. Acest numar indica pe citi biti este codificata culoarea la nivelul fiecarui pixel. In cazul utilizarii a 24 de biti/pixel nu este necesra utilizarea unei tabele de culori, fiecare pixel fiind codificat printr-o structura de trei octeti:

```
typedef struct{
        byte      Blue;     /* intensitatea componentei albastre    */
        byte      Green;    /* intensitatea componentei verzi       */
        byte      Red;      /* intensitatea componentei rosii       */
}RGB;
```

In celelalte cazuri se utilizeaza o tabela de culori care este utila pentru a asocia o culoare fiecarui cod de 1, 4 sau 8 biti folosit pentru reprezentarea culorii. Astfel, in cazul utilizarii a 8 biti/pixel rezulta ca avem la dispozitie 256 de coduri, deci culori diferite. Pentru a indica ce culoare corespunde ficarui cod se utilizeaza o tabela cu 256 de elemente cu urmatoarea stuctura:

```
typedef struct{
        byte      Blue;     /* intensitatea componentei albastre    */
        byte      Green;    /* intensitatea componentei verzi       */
        byte      Red;      /* intensitatea componentei rosii       */
        byte      Res;      /* rezervat (=0)                        */
}RGBQUAD;
```

Codul de 1, 4 sau 8 biti reprezinta indexul in aceasta tabela de unde se citeste culoarea reala a acelui pixel. Parametrul ClrUsed specifica numarul de elemente ale tabelei de culori (daca numarul de elemente este egal cu numarul maxim de coduri reprezentabile pe numarul de biti utilizat, atunci valoarea acestui parametru este 0). Parametrul ClrImp defineste numarul de elemente din tabela utilizate efectiv in imagine (daca sint utilizate toate, atunci acest parametru are valoarea 0).

Parametrul Comp specifica tipul de compresie utilizat. Desi, initial, se prevazuse posibilitatea utilizarii unor mecanisme de compresie, in momentul actual pentru aceste situatii s-au definit alte formate astfel incit acest parametru va vi sigur egal cu 0 adica nu se utilizeaza compresie.

Parametrii XDef si YDef se refera la dimensiunea fizica a unui pixel (orizontala si verticala). Astfel, daca acesti parametrii au valoarea 2000 insemna ca un pixel are dimensiunea de 0.5x0.5mm. Acest lucru este valabil numai in cazul prezentarii imaginii pe medii care permit impunerea unei anumite dimensiuni pentru pixeli (cazul tiparirii imaginii) si nu in cazul afisarii pe ecran.

Ultima sectiune este cea care contine codificarea propriu-zisa a imaginii.

Aceasta sectiune contine codurile care corespund culorilor fiecarui pixel daca imaginea este parcursa linie cu linie de la ultima (cea mai de jos) catre prima (cea mai de sus), fiecare linie fiind parcursa de la stinga la dreapta.

Codurile sint impachetate la nivel de octet (de exemplu daca se utilizeaza 1 bit/pixel fiecare octet va contine 8 pixeli).

Codificarea unei linii trebuie sa contina un numar de octeti multiplu de 2 in cazul utilizarii codiificarii de 1 bit/pixel sau multiplu de 4 in celelalte cazuri, acest lucru realizindu-se prin completarea liniei cu un numar corespunzator de octeti suplimentari egali cu 0.

Compresia imaginilor

In continuare vom prezenta citeva tehnici de compresie fara a avea pretentia ca sint cele mai bune sau conforme cu un standard (desi prima este inspirata din standardul JPEG).

Inainte, insa, vreau sa precizez ca exista doua categorii principale de metode si anume:

- **metode fara pierdere de informatie** - la care procesul de compresie nu altereaza sub nici o forma informatia, adica imaginea comprimata este exact la fel cu cea necomprimata. Aceste metode nu permit insa obtinerea unei rate de compresie (raport intre numarul de octeti ai imaginii necomprimate si cel al celei comprimate) mai bun de 2:1 rar 3:1.
- **metode cu pierdere de informatie** - in cazul carora pentru obtinerea unei rate de compresie mai bune (se poate ajunge si la 50:1 sau mai mai mult) se accepta o pierdere de informatie, deci o alterare a imaginii cu conditia ca acest lucru sa fie cit mai putin vizibil.

http://www.indinf.pub.ro/catalinp/img/img.htm

Metoda pe baza transformatei cosinus discrete

Aceasta metoda este inspirata din procedura de compresie JPEG si face parte din categoria celor cu pierdere de informatie. Procedura de compresie este data pentru codificarea unei imagini monocrome (cu o singura componente de culoare reprezentata de regula pe 8 biti) de dimensiune 8x8 pixeli. In cazul unei imagini care are alta dimensiune (8x8 pixeli este cam putin) se procedeaza la impartirea acesteia in blocuri avind aceasta dimensiune. Pentru compresia imaginilor color se procedeaza la compresia independenta, dupa aceasta procedura a celor trei componente (RGB sau YUV) pentru fiecare bloc din imagine.

Un bloc de 8x8 pixeli poate fi considerat o matrice de 8x8 elemente, fiecare element fiind un numar intreg reprezentat pe un numar oarecare de biti (de obicei 8) si care codifica intensitatea componentei careia ii corespunde acel bloc. In cazul in care se utilizeaza reprezentarea RGB valorile elementelor sint intotdeauna pozitive iar daca se utilizeaza reprezentarea YUV atunci elementele corespunzatoare blocurilor componentei Y sint pozitive iar cele ale componentelor U si V pot lua si valori negative.

… Metoda pe baza transformatei cosinus discrete

In general, pentru compresie, imaginile sunt convertite in reprezentare YUV.

Daca analizam formulele de calcul care descriu aceasta conversie se poate preciza domeniul de variatie al celor trei componente pentru o reprezentare RGB pe 8 biti:

$$Y = 0.30*R + 0.59*G + 0.11*B$$

$$U = R - Y = 0.70*R - 0.59*G - 0.11*B$$

$$V = B - Y = -0.30*R - 0.59*G + 0.89*B$$

In cazul acestor relatii se obtin pentru toate cele trei componente Y, U si V acelasi domeniu de variatie si anume -128..+128.

… Metoda pe baza transformatei cosinus discrete

Componentele R,G si B fiind codificate pe 8 biti pot lua valori intregi in domeniul 0..255. Componenta Y, la rindul ei va avea valori intre 0..255 deoarece 0.30+0.59+0.11=1.

Componentele U si V, insa, pot varia intre -179..+179 (0.7x255) respectiv intre -227..+227 (0.89x255).

Se observa o neuniformitate in aceste domenii de valori care poate ridica probleme in estimarea preciziei si cerintelor de reprezentare a informatiei pe parcursul procesului de compresie.

Din acest motiv propun un alt set de relatii pentru conversia din RGB in YUV si anume:

$$Y = 0.30*R + 0.59*G + 0.11*B - 128$$

$$U = 0.50*R - 0.42*G - 0.08*B$$

$$V = -0.17*R - 0.33*G + 0.50*B$$

… Metoda pe baza transformatei cosinus discrete

Prima operatie care se realizeaza in scopul compresiei este transformarea "imaginii" continute in matrice (bloc) intr-o forma echivalenta.

In acest scop este utilizata transformata cosinus discreta care pornind de la matricea imagine S genereaza o matrice transformata T dupa urmatoarea formula de calcul (in care se considera cazul matricilor S respectiv T de 8x8 elemente):

$$T_{i,j} = \frac{1}{4} C_i C_j \sum_{x=0}^{7} \sum_{y=0}^{7} S_{y,x} \cos\frac{(2x+1)j\pi}{16} \cos\frac{(2y+1)i\pi}{16}$$

… Metoda pe baza transformatei cosinus discrete

Trecerea inversa de la matricea transformata la matricea imagine initiala se poate realiza prin transformata inversa descrisa de relatia:

$$S_{i,j} = \frac{1}{4} \sum_{x=0}^{7} \sum_{y=0}^{7} C_y C_x T_{y,x} \cos \frac{(2j+1)x\pi}{16} \cos \frac{(2i+1)y\pi}{16}$$

In ambele relatii avem:

$C_i, C_j = \frac{1}{\sqrt{2}}$, pentru i,j = 0 , altfel, $C_i, C_j = 1$

… Metoda pe baza transformatei cosinus discrete

Elementele matricii transformate reprezinta de fapt ponderile cu care se insumeaza un numar de 64 de matrici "sablon" constante pentru a se obtine matricea imagine.

Aceste matrici sablon pot fi imaginate ca niste imagini de tip "tabla de sah", elementului (i,j) din T corespunzindu-i o tabla ci i linii si j coloane. Deci, se poate afirma ca elementele matricii transformate cu indici mici (i sau j) corespund detaliilor mai grosiere din imagine iar cele cu indici mai mari detaliilor mai fine.

Din acest motiv, daca analizam matricile T obtinute prin transformarea diferitelor blocuri dintr-o imagine, vom observa ca in coltul din stinga-sus al matricilor avem valori mari (pozitive sau negative) si cu cit coborim spre coltul din dreapta-jos valorile elemntelor incep sa scada multe chiar spre valoarea zero.

O precizare in legatura cu transformata cosinus discreta este aceea ca pentru o matrice S cu elemente in intervalul -128..+128 se pot obtine in matricea T valori sigur cuprinse intre -1023..1023.

… Metoda pe baza transformatei cosinus discrete

Transformarea imaginii este o operatie care teoretic nu altereaza imaginea. Se observa ca exista o transformata inversa exacta, singura posibilitate de alterare a imaginii fiind erorile de calcul (mai ales cind se utilizeaza calcul in virgula fixa). Pentru compresie, insa este necesar sa se piarda ceva informatie. Acest lucru se poate realiza printr-o reducere a preciziei de reprezentare a elementelor matricii transformate.

O modalitate de reducere a preciziei este aceea de a imparti acel numar cu o anumita valoare si de a retine doar partea intreaga din rezultat. Refacerea valorii initiale (bineinteles aproximativ)se realizeaza foarte simplu inmultind numarul intreg obtinut cu valoarea cu care s-a realizat impartirea. Aceasta operatie se numeste cuantizare.

Problema este cu cit sa se imparta elementele matricii transformate astfel incit la refacerea imaginii efectul acestei aproximatii sa fie cit mai putin vizibil.

… Metoda pe baza transformatei cosinus discrete

Pentru a rezolva aceasta problema se porneste de la observatia ca detaliile fine au o importanta mai redusa in aprecierea calitatii unei imagini iar aceste detalii, asa cum am aratat mai sus, sint caracterizate de elementele din matricea transformata situate catre coltul din dreapta-jos. Astfel se poate defini o matrice de cuantizare care sa contina in fiecare pozitie valoarea prin care se va realiza impartirea elementului corespunzator din matricea transformata. Aceasta matrice poate fi aleasa astfel incit sa fie simetrica si elementele sale sa creasca pe masura ce se coboara din coltul din stinga-sus catre cel din dreapta-jos. Un exemplu de matrice este dat in continuare:

```
int q[8][8]={    { 4, 4, 5, 5, 6, 6, 7, 7},
                 { 4, 5, 5, 6, 6, 7, 7, 8},
                 { 5, 5, 6, 6, 7, 8, 9,10},
                 { 5, 6, 6, 7, 8, 9,10,11},
                 { 6, 6, 7, 8,10,12,14,16},
                 { 6, 7, 8, 9,12,13,16,19},
                 { 7, 7, 9,10,14,16,20,24},
                 { 7, 8,10,11,16,19,24,29} };
```

… Metoda pe baza transformatei cosinus discrete

Prin operatia de cuantizare se obtine o matrice H in care descresterea valorii elementelor este mult mai pronuntata, multe elemente devenind zero.

Un exemplu de matrice obtinuta prin cuantizare este cel prezentat mai jos:

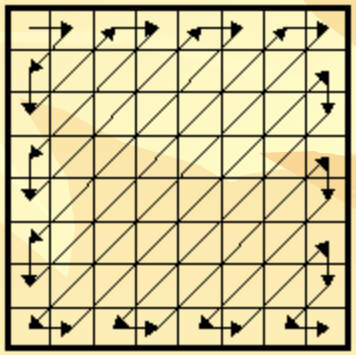
236	-105	-24	11	0	0	0	0
-13	-17	0	0	0	0	0	0
11	0	0	0	0	0	0	0
0	0	0	-3	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Urmatoarea etapa este cea de codificare propriu-zisa a acestor elemente din matrice.

Dupa cum se observa in exemplul de mai sus valoarea predominanta este zero iar valorile mici sint cele mai frecvente. Deasemenea valorile nule se concentreaza in coltul din dreapta-jos a matricii.

… Metoda pe baza transformatei cosinus discrete

O prima operatie care poate fi realizata in vederea codificarii este a ordona elementele din matrice intr-un sir astfel incit sa se puna in evidenta siruri cit mai lungi de valori nule. Ordonarea se face citind matricea in zig-zag conform figurii urmatoare:



Se obtine, astfel, pentru exemplul de mai sus urmatorul sir de valori:

```
236,-105,-13,11,-17,-24,11,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,-13,
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
```

Pentru codificarea acestui sir de valori se utilizeaza o metoda bazata pe asocierea cite unui cod pentru siruri de elemente de tipul:

```
0,0...0,N - Z elemente egale cu zero plus un element diferit de zero
```

… Metoda pe baza transformatei cosinus discrete

Bineinteles, pot exista situatii cind intre doua elemente succesive nu exista nici o valoare zero, caz in care numarul de valori zero (Z) va fi nul.

Valoarea elementului diferit de zero (N) va fi codificata prin trei elemente, de exemplu, valoarea -13 va fi caracterizata prin semn (-), numar de biti semnificativi (4) si bitii propriu-zisi (1101).

Se face o observatie in legatura cu aceasta reprezentare si anume, intotdeauna bitii propriu-zisi incep printr-un bit 1 deoarece daca ar fi zero atunci am avea un numar de biti semnificativi mai mic.

Din acest motiv nu este necesar sa includem acest bit in codificare el fiind implicit 1 deci pentru valoarera -13 se retin doar (101).

… Metoda pe baza transformatei cosinus discrete

Numarul Z si numarul de biti semnificativi vor primi impreuna un cod (clasa) iar semnul lui N si bitii din care este compusa valoarea acestuia vor fi atasati dupa acest cod.

Daca utilizam pentru codul unei clase notatia (Z,B) unde B reprezinta numarul de biti semnificativi si facem conventia ca semnul (-) se reprezinta printr-un bit 1 iar (+) print-un bit 0, atunci codificarea exemplului de mai sus este urmatoarea (bitul marcat intre paranteze este cel implicit care nu trebuie codificat):

clasa	semn	biti
(0,8)	0	(1)1101100
(0,7)	1	(1)101001
(0,4)	1	(1)101
(0,4)	0	(1)011
(0,5)	1	(1)0001
(0,5)	1	(1)1000
(0,4)	0	(1)011
(17,2)	1	(1)1
(39,0)		

… Metoda pe baza transformatei cosinus discrete

Ultima clasa codificata contine ca indicator de numar de biti semnificativi pentru elementul diferit de zero valoarea zero deoarece nu exista acest element (sirul de elemente se termina in zerouri).

Urmatorul pas este de a gasi coduri pentru clase.

Prima problema este de a determina numarul de clase posibile.

Pentru acesta se observa ca numarul de zerouri nu poare depasi 64 (nu exista mai mult de 64 de elemente in sir), deci pentru Z exista 65 de valori posibile. Apoi din cele prezentate despre transformata cosinus discreta elementele matricii transformate sint in intervalul -1024..1024, deci pot avea maximum 10 biti semnificativi (plus semn). Daca se tine cont ca valorile care se codifica sint valorile matricii transformate dupa cuantizare (deci dupa impartirea acestora cu ceva), atunci sigur B nu poatr depasi 10.

Exista, deci 11 valori posibile pentru B si 65 de valori posibile pentru Z ceea ce conduce la un numar total de clase egal cu 65x11=715.

… Metoda pe baza transformatei cosinus discrete

In aceste conditii o clasa poate fi codificata printr-un numari intreg pozitiv pe 10 biti obtinut dupa o relatie de tipul CLASA=65xB+Z.

In acest caz, exemplul de mai sus are codificarea binara urmatoare:

cod clasa	semn	biti
1000001000	0	1101100
0111000111	1	101001
0100000100	1	101
0100000100	0	011
0101000101	1	0001
0101000101	1	1000
0100000100	0	011
0010010011	1	1
0000100111		

… Metoda pe baza transformatei cosinus discrete

Se constata ca sint necesari 129 de biti fata de 64x8=512 biti citi ar fi fost necesari in varianta necomprimata.

Rezulta o compresie de 3.97:1 ceea ce nu reprezinta prea mult.

Pentru o imbunatatire suplimentara se introduce o modificare in modalitatea de codificare a claselor si anume nu se mai utilizeaza un cod cu lungime fixa ci unul cu lungime variabila de tip Huffman.

Acest lucru conduce la o reducere a numarului de biti utilizati pentru codificarea claselor deoarece nu toate clasele au aceesi frecventa de aparitie (o clasa de tip (26,9) adica un numar pe 9 biti dupa 26 de zerouri este aproape imposibila). Codul Huffman aloca simbolurilor (in cazul acesta claselor) cele mai frecvente coduri mai scurte iar celor mai putin frecvente coduri mai lungi.

Compresia imaginilor

Compresia imaginilor

martie 22, 2008 de Andrei Dumitrescu

Marimea unei imagini poate fi mai mare decat marimea unui fisier normal, de exemplu o imagine cu rezolutia de 800×600 are 480.000 de pixeli, fiecare pixel foloseste 24 de biti (True Color) adica 3 bytes, deci fisierul poate atinge 1,44 Mb. Compresia imaginilor a fost inventata pentru a reduce marimea fisierelor, pentru a economisii spatiu pe mediul de stocare, si nu in ultimul rand pentru a le face mai usor de manevrat.

In timpul compresiei imaginii, datele care apar de mai multe ori sau cele fara valoare sunt eliminate sau salvate intr-o forma mai scurta, astfel reducandu-se marimea fisierului. Cand imaginea este afisata atunci se porneste un proces invers compresiei. Exista doua tipuri de compresie, fara pierdere de calitate si cu pierdere de calitate.

http://artafotografica.wordpress.com/2008/03/22/compresia-imaginilor/

Compresia fara pierdere de calitate

Suna bine, nu? Acesti algoritmi urmaresc ca fisierul rezultat sa fie la acelasi nivel calitativ cu fisierul initial. Cel mai performant algoritm este LZW (Lempel-Ziv-Welch). Acesta este folosit in fisierele TIFF si GIF, si atinge rate de compresie de la 50% la 90%.

Compresia cu pierdere de calitate

Acest tip de compresie este folosit de marea majoritate a aparatelor foto digitale, si practic imaginea este comprimata pana la un anumit nivel, cu cat comprimarea este mai mare, cu atat imaginea este mai degradata. In mod normal o comprimare medie nu deranjeaza pe nimeni si nu se observa mari diferente, numai daca imaginile sunt marite. Aceasta compresie este deosebit de buna pentru imaginile care vor fi puse pe internet. Secretul compresiei este acela de a inlatura informatile care nu sunt evidente ochiului uman, daca intr-o poza avem o zona de o anumita culoare, atunci este memorat doar un pixel cu informatia sa despre culoarea respectiva.

Cel mai bun algoritm este **JPEG** (Joint Photographic Experts Group), algoritm ce permite selectarea gradului de compresie. In general se folosesc rate de compresie de la 10:1 la 40:1.

In memoria aparatelor, imaginile sunt stocate in anumite formate. La ora actuala exista o multime de formate pentru a stoca imaginile insa exista programe cu ajutorul carora puteti converti fisierele dintr-un format in altul. Sa vedem care sunt cele mai comune formate: **RAW, TIFF, JPEG.**

RAW

Compresia poate degrada intr-o anumita proportie imaginea, de aceea unele aparate permit stocarea lor necomprimate pentru a pastra o calitate cat mai mare posibil. In unele cazuri aceste fisiere sunt proprietatea unor firme si nu sunt prezente in alte aplicatii decat cele dezoltate de firma respectiva. De exemplu Canon foloseste fisiere necomprimate care interpoleaza imaginea o data ce este descarcata pe computer. Alte aparate folosesc un format mai universal – TIFF.

TIFF

TIFF (Tag Image File Format) a fost initial descoperit si dezvoltat de corporatia Aldus pentru a stoca imaginile create de scannere si programe de editare grafica. Acest format este foarte raspandit, mai ales in aplicatile DTP. Acest format de imagine are mai multe variante, unele pot fi comprimate cu algoritmi fara pierdere de imagine, precum LZW. Aceste fisiere suporta o adancime a culorii de pana la 24 de biti.

JPEG

JPEG (Joint Photographic Experts Group) este formatul de imagine cel mai des folosit pentru stocarea imaginilor si/sau afisarea lor pe internet. Acest algoritm se bazeaza pe faptul ca oamenii sunt mult mai senzitivi la micile modificari de stralucire (luminescenta) decat la cele de culoare.

Acest format suporta adancimi de culoare de pana la 24 de biti. Este bine ca sa alegeti acest format de imagine atunci cand ea nu va mai fi editata, si acesta este rezultatul final. Cat timp editati imaginea, folositi formate fara pierdere de calitate precum BMP si TIFF. Compresia este realizata in blocuri de 8×8 pixeli, care sunt vizibili daca folositi un grad mare de compresie si mariti imaginea.

Algoritmi

1. Algoritmul Huffman
2. Algoritmul RLE
   a) Algoritmul RLE pentru imagini alb/negru
   b) Algoritmul RLE pentru imagini cu nuante de gri
3. Compresia cu transformate

http://www.miv.ro/ro/documentatie/pi/PIlab09.pdf

Bibliografie

1. http://artafotografica.wordpress.com/2008/03/22/compresia-imaginilor/
2. http://www.miv.ro/ro/documentatie/pi/PIlab09.pdf
3. http://193.226.17.10/romana/Documente%20Discipline/CCISV/Lucrari%20laborator/L4.pdf
4. http://facultate.regielive.ro/proiecte/electronica/compresia_imaginilor-29236.html
5. http://facultate.regielive.ro/cursuri/electronica/compresia_imaginilor_binare-3686.html
6. http://www.studentie.ro/Tutoriale/Tutoriale-foto/Compresia-imaginilor_i75_c951_70951.html
7. http://www.ginfo.ro/revista/13_5/serial.pdf
8. http://www.onlinestudent.ro/referate/914243f66efa2c6ccc774176843e96e7.doc
9. http://shannon.etc.upt.ro/docs/anIV/jpeg.pdf
10. http://openpdf.com/ebook/compresia-pdf.html
11. http://www.miv.ro/ro/documentatie/pi/PIlab09.pdf
12. Mihai Ciuc, Constantin Vertan - *"Prelucrarea Statistică a Semnalelor"*, MatrixROM, Bucuresti, 2005[Download PDF]
13. Constantin Vertan, Inge Gavăt, Rodica Stoian - *"Variabile şi Procese Aleatoare: principii şi aplicaţii"*, Bucureşti, 1999[Download PDF]
14. Mihai Ciuc - *"Fourier"*[Download PDF]
15. Mihai Ivanovici - *"Procesarea Imaginilor - îndrumar de laborator"*, Universitatea Transilvania, Braşov, 2006[Download PDF]