PRELUCRAREA ŞI ANALIZA IMAGINILOR

Constantin VERTAN

{1999}

Cuprins

Capitolul 1

INTRODUCERE

Prelucrarea şi analiza imaginilor (numită adeseori prescurtat doar prelucrarea imaginilor) s-a născut datorită ideii şi necesităţii de a înlocui observatorul uman printr-o maşină. Este important de precizat că analiza imaginilor a mers mai departe decât simpla înlocuire a observatorului uman, deoarece au apărut soluţii novatoare pentru probleme cu care acesta nu mai fusese confruntat - ca în cazul imaginilor non-vizibile (imagini acustice, ultrasonore, radar). După cum se remarcă în [9], prelucrarea imaginilor înglobează posibilitatea de a dezvolta maşina totală de viziune, capabilă să realizeze funcţiile vizuale ale oricărei vieţuitoare (desigur, după realizarea a importante dezvoltări teoretice şi tehnologice).

> "Image processing holds the possibility of developing the ultimate machine that could perform the visual functions of all living beings".

Trebuie remarcată terminologia anglo-saxonă (originală), în care disciplina este denumită *Digital Image Processing*, deci prelucrarea digitală a imaginilor. Prin prelucrarea digitală a imaginilor se înţelege prelucrarea pe un calculator digital a unor date bidimensionale (imagini). Termenul cheie este cuvântul *digital*, înlocuit adesea în mod eronat în multe traduceri româneşti cu termenul de numeric. După cum o arată dicţionarul limbii române moderne, definiţia cuvântului *numeric* este aceea de

> "care aparţine numerelor, privitor la numere, exprimat prin numere".

Rezultatul oricărui calcul este numeric. Termenul *digital* înseamnă însă

> "ceea ce este referitor la reprezentarea informaţiei discrete în calculatoare"

Deci atâta vreme cât acceptăm ideea că unealta de lucru în prelucrarea imaginilor este calculatorul, şi acesta la rândul său este digital, atunci şi prelucrarea este la rândul ei digitală, ca un caz particular al oricărei prelucrări numerice. Desigur că există însă şi prelucrări de imagini care sunt analogice - aşa cum sunt toate prelucrările ce au loc în cadrul lanţului de transmisie şi recepţie a imaginii standard de televiziune.

1.1 Imagini digitale

La început, imaginile sunt semnale, dar nu funcţii temporale, ci funcţii definite pe un domeniu spaţial. Orice imagine este o structură bidimensională (tablou, matrice) de date. Un element al imagini se numeşte *pixel* (cuvânt preluat din engleză, unde provine de la **pic**ture **el**ement). Aceste date pot fi numere naturale, reale sau complexe, reprezentate însă pe un număr finit de biţi. După tipul datelor din acestă structură bidimensională, imaginile prelucrate pot fi împărţite în mai multe categorii:

- imagini scalare, în care fiecare componentă este un scalar (un unic număr); ca exemple de astfel de imagini se pot da imaginile monocrome (în care punctele au doar două valori posibile, ce corespund unui conţinut binar al imaginii, în general alb-negru) şi imaginile cu nivele de gri (de tipul imaginii de luminanţă de pe ecranele televizoarelor alb-negru).
- imagini vectoriale, în care fiecare componentă este un vector de numere; cazul particular cel mai de interes este acela al imaginilor color, în care vectorul are trei elemente (ce corespund celor trei constituente de bază ale oricărei culori); în general, pentru imaginile multicomponentă, vectorul asociat fiecărui punct din imagine are mai multe elemente (caz ce corespunde imaginilor preluate în mai multe benzi de frecvenţă, aşa cum sunt imaginile de teledetecţie ale sateliţilor, imaginile de termodetecţie în benzile de infraroşu,...). Tot în categoria imaginilor vectoriale intră însă şi imaginile stereo (o pereche de imagini ale aceleiaşi scene, luate din unghiuri diferite) şi secvenţele de imagini.

Conform datelor prezentate în [11], dintre imaginile prelucrate în aplicaţii funcţionale, 20 % sunt alb-negru, 32 % sunt cu nivele de gri, 20 % sunt color, 10 % sunt imagini stereoscopice şi 18 % sunt secvenţe de imagini.

În mod clasic, valoarea unui element al unei imagini este o măsură a intensităţii luminoase în punctul considerat; acesta nu este însă decât un caz particular. După natura lor, imaginile pot fi clasificate ca imagini abstracte, imagini non-vizibile şi imagini vizibile [2]. Imaginile abstracte sau modelele sunt de fapt funcţii [matematice], continue sau discrete, de două variabile. Imaginile non-vizibile, care, evident, nu pot fi percepute în mod direct de ochiul uman, sunt de fapt achiziţii ale unor câmpuri bidimensionale de parametri fizici

(presiune, temperatură, presiune, densitate, ...). În fine, imaginile ce pot fi percepute în mod direct de către ochiul uman (deci imaginile vizibile) sunt la rândul lor imagini optice, generate ca distribuţii de intensitate luminoasă (aşa ca hologramele, imaginile de interferenţă şi difracţie) sau imagini propriu-zise (de luminanţă - în sensul curent al termenului, ce se referă la fotografii, desene, picturi, schiţe, scheme şi altele din aceeaşi categorie).

O altă împărţire a imaginilor scalare se poate face după semnificaţia ce se dă valorii numerice a pixelilor. Vom distinge astfel imagini de intensitate şi imagini indexate. O imagine de intensitate este o imagine în care valoarea fiecărui pixel este o măsură directă a intensităţii luminoase sau a mărimii fizice preluate de senzor, ca de exemplu în imaginile cu nivele de gri. Pixelii unei imagini de intensitate pot avea orice fel de valori: reale sau naturale (depinzând dacă imaginea este sau nu cuantizată).

O imagine indexată este acea imagine în care valoarea fiecărui pixel este un indice prin care se regăseşte informaţia de culoare asociată pixelului respectiv. Deci, pentru afişarea sau reprezentarea unei imagini indexate este necesară o informaţie suplimentară, de asociere între indici şi culori. Această asociere se face prin intermediul tabelei de culoare. Tabela de culoare este o matrice în care fiecare linie conţine descrierea unei culori (deci cele trei componente ce definesc culoarea - în mod tipic intensităţile relative de roşu, verde şi albastru ce compun culoarea dată printr-un amestec aditiv). Deci tabela de culoare are trei coloane; numărul de linii al tabelei de culoare este egal cu numărul de culori din imaginea reprezentată şi este în mod tipic o putere a lui doi (16, 256, ...). Indicele (valoarea pixelului) va fi numărul de ordine al liniei din tabela de culoare pe care se găseşte descrierea culorii. Este evident că valorile pixelilor unei imagini indexate nu pot fi decât numere naturale (deoarece sunt indici într-o matrice).

Această tehnică este folosită şi în grafica pe calculator. Afişarea imaginilor pe ecranul monitorului se face corespunzător unui anumit mod grafic, determinat din placa video a calculatorului. Un mod video defineşte numărul maxim de culori ce pot fi utilizate simultan şi dimensiunile ecranului (în pixeli de afişaj). Culorile utilizate la un moment dat sunt grupate într-o paletă de culori de afişare. Paleta de afişare este o structură logică definită în BGI[1] (Borland Graphics Interface), pentru programare în sesiuni de tip DOS, ca:

```
struct palettetype {
  unsigned char size;
  int colors[MAXCOLORS+1]; }
```

Modificarea unei culori din paletă (o intrare a tabelului) se face cu:

```
void far setpalette(int index_culoare, int culoare);
```

[1]Exemplele de cod C prezentate în lucrare corespund mediilor integrate Borland (Borland C++ 3.1, Turbo C 2.0)

Afişarea unui pixel cu o anumită culoare se face cu:

```
putpixel(int pozx, int pozy, int index_culoare);
```

Sub Windows, este suportată şi specificarea directă a culorii de afişat (sub forma unui triplet RGB[2]), sistemul de operare aproximând culoarea respectivă cu cea mai apropiată culoare disponibilă din paleta de lucru curentă (în acest fel, utilizatorul poate neglija existenţa acesteia).

Pentru o imagine cu nivele de gri, componentele de roşu, verde şi albastru ale fiecărei culori din paleta de culoare sunt identice. Dacă specificarea componentelor de culoare se face prin numere de 8 biţi (deci între 0 şi 255, adică cazul cel mai des folosit), tabela de culoare va avea 256 de culori (tonuri de gri) diferite. Specificarea acestora se va face cu indecşi între 0 şi 255, alocaţi conform convenţiei 0 - negru, 255 - alb. În acest fel, pentru o imagine indexată cu nivele de gri, nu mai este necesară specificarea tabelei de culoare; culorii reprezentate de indexul i îi corespunde nivelul de gri i, adică tripletul RGB (i, i, i).

Modelul imagini indexate este un caz particular de folosire a tehnicii dicţionar (sau tehnicii tabelului de echivalenţă - *Look Up Table* - LUT): o tehnică de regăsire a unei cantităţi de informaţie folosind asocierea unei chei de căutare mult mai mici.

1.2 Structura unui sistem de prelucrarea şi analiza imaginilor

Structura tipică a unui sistem de prelucrarea (evident digitală) şi analiza imaginilor este alcătuită din punct de vedere funcţional dintr-un număr mic de blocuri (vezi figura 1.1):

- sistemul de formare a imaginii (de exemplu sistemul de lentile al camerelor de luat vederi): strânge radiaţia electromagnetică a obiectului studiat pentru a forma imaginea trăsăturilor de interes
- convertorul de radiţie: converteşte radiaţia electromagnetică din planul imaginii într-un semnal electric.

Sistemul de formare a imaginii şi convertorul de radiaţie formează senzorul; acesta realizează o proiecţie plană (bidimensională) a scenei reale (care este în general tridimensională). Un studiu realizat în Germania în anul 1996 [11] prin inventarierea sistemelor de

[2]Red Green Blue - Roşu, Verde, Albastru: sistemul primar de reprezentare a culorilor.

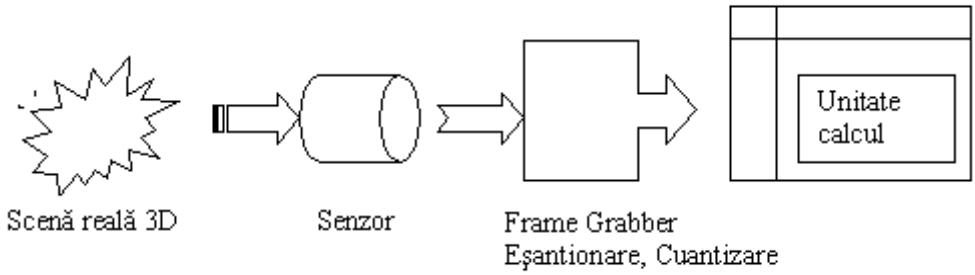

Fig. 1.1: Schema generală a unui sistem de analiza şi prelucrarea imaginilor

preluare a imaginilor folosite în industrie indică o distribuţie a tipurilor de senzori după gama de radiaţie captată conform tabelului 1.1:

Domeniu electromagnetic	Infraroşu depărtat	Infraroşu apropiat	Vizibil	Ultraviolet	Radar (microunde)	Radiaţie X
Procent	5 %	25 %	40 %	10 %	10 %	10 %

Tabel 1.1: Tipuri de senzori folosiţi în prelucrarea imaginilor

- sistemul de achiziţie (echivalent unui *frame-grabber* sau *video-blaster*): converteşte semnalul electric al senzorului într-o imagine digitală, pe care o stochează; acesta nu este altceva decât un dispozitiv de eşantionare (discretizare a suportului imaginii) şi cuantizare (discretizare a domeniului de valori a imaginii).
- sistemul de prelucrare (în mod tipic un calculator, fie el PC sau staţie de lucru); în această categorie se încadrează însă şi maşinile specializate de calcul, calculatoarele de proces, ...
- software-ul specializat: implementează algoritmii de prelucrare şi analiză

În [11] se arată că unitatea de prelucrarea hardware (deci calculatorul) folosită la aplicaţiile de prelucrarea imaginilor funcţionale la acea dată este în cea mai mare majoritate a cazurilor un PC obişnuit, cu performanţe standard; datele sunt sintetizate în tabelul 1.2:

Platformă hardware	Procent din piaţă
PC standard, bus ISA, Windows 3.1, 95, NT	40 %
Calculatoare industriale (procesoare Intel), bus VME	15 %
PC standard cu acceleratoare specializate, bus VLB, PCI	15 %
Staţii de lucru (workstations)	10 %
Maşini specializate	10 %
Calculatoare Macintosh, calculatoare paralele (transputere), altele	10 %

Tabel 1.2: Unităţi de calcul folosite în prelucrarea imaginilor

Sistemul software specializat care este responsabil cu realizarea efectivă a unei sarcini concrete poate fi descompus în mai multe module, nu neapărat bine separate şi nu neapărat prezente împreună: îmbunătăţirea, restaurarea, compresia, segmentarea şi analiza [9].

Blocul de îmbunătăţire a imaginilor are ca scop accentuarea anumitor trăsături [ale obiectelor conţinute în imagine] pentru uşurarea unor sarcini ulterioare de analiză automată sau interpretare prin afişare. Asemenea metode pot fi utile la extragerea trăsăturilor caracteristice ale obiectelor, eliminarea zgomotelor suprapuse imaginii, mărirea confortului vizual. Aceşti algoritmi nu măresc conţinutul informaţional al imaginii şi sunt în general interactivi şi puternic dependenţi de aplicaţie.

Restaurarea imaginilor se referă la eliminarea sau minimizarea efectelor unor perturbaţii şi a unor degradări. Perturbaţiile reprezintă în general zgomotele (modelate ca procese aleatoare) ce se suprapun în cursul achiziţiei imaginii (din cauza senzorului şi a lanţului de transmisiune şi captare); degradările sunt cauzate de imperfecţiunile şi limitările deterministe ale senzorului (efecte de apertură, timp de expunere, deficienţe geometrice ale sistemului de lentile, ...).

Compresia imaginilor se referă la reducerea volumului de date (numărului de biţi) cu care este reprezentată imaginea, printr-o transformare reversibilă - imaginea trebuie să poată să fie recuperată integral (sau cu diferenţe foarte mici, controlabile) din versiunea sa comprimată.

Segmentarea este procesul de descompunere a unei imagini (sau scene) în elementele (obiectele) sale constituente. Adeseori, segmentarea este strâns legată de algoritmii de analiză, al căror scop este de a realiza măsurători cantitative sau evaluări calitative asupra unor anumite categorii de obiecte, prezente în imaginea dată.

Sfera de aplicabilitate a tehnicilor de prelucrarea şi analiza imaginilor este deosebit de largă; practic, în orice domeniu de activitate se pot găsi numeroase aplicaţii. Această clasă de aplicaţii extrem de specifice a fost caracterizată drept “consumul imaginii” [1] (imaginea folosită în vederea analizei, deci a luării unor decizii). Imaginile preluate de către sateliţi pot fi folosite la descoperirea resurselor terestre, cartografiere geografică, predicţia recoltelor, urmărirea dezvoltării urbane, urmărirea vremii, controlul şi prevenirea incendi-

ilor şi inundaţiilor, precum şi alte aplicaţii ecologice şi militare. Aplicaţiile transmisiei şi compresiei imaginilor se regăsesc în televiziunea digitală, sistemele de teleconferinţă, transmisiile fax, birotică, comunicaţia pe reţele distribuite, sisteme de securitate cu circuit închis, aplicaţii militare. În aplicaţiile medicale sunt prelucrate radiografiile cu raze X, angiogramele, echografiile, tomografiile, imaginile de rezonanţă magnetică nucleară. Acestea pot fi utilizate pentru monitorizarea pacienţilor şi pentru descoperirea şi identificarea de boli şi tumori.

O largă clasă de aplicaţii sunt cele industriale, în care componentele de prelucrarea şi analiza imaginilor sunt folosite în sisteme mai mari de asigurare a calităţii produselor (metrologie, controlul calităţii - inclusiv defectoscopie nedistructivă). Soluţiile sunt extrem de specifice, puternic legate de procesul de fabricaţie urmărit şi tind să devină din ce în ce mai utilizate odată cu impunerea normelor de "calitate totală" ale standardului ISO9000 (se poate urmări [10] pentru aplicaţii specifice ale diferitelor firme germane). Din acest punct de vedere este interesantă comparaţia între câteva caracteristici ale sistemului vizual şi de prelucrare uman şi un sistem de prelucrarea şi analiza imaginilor [8], folosite pentru aplicaţii industriale, prezentată în tabelul 1.3.

Criterii	Om	Sistem de prelucrarea imaginilor
Obiectivitate	NU	DA
Control 100%	NU	DA
Rată de eroare	MARE	MICĂ
Rată de lucru	MICĂ	MARE
Rezistenţă la oboseală	MICĂ	MARE
Iluzie optică	DA	NU
Prelucrare statistică	Greu realizabil	DA
Reproductibilitate	Greu realizabil	DA
Măsurare geometrică	Cu instrumente auxiliare	DA
Recunoaştere de forme	DA	DA

Tabel 1.3: Comparaţia între caracteristici esenţiale ale sistemului vizual uman şi sistemele de prelucrarea şi analiza imaginilor

1.3 Stocarea imaginilor

Se poate considera că există două moduri de stocare a imaginilor: stocarea în memoria de lucru a unităţii de prelucrare a imaginii de lucru (care este o stocare de scurtă durată - doar pe durata prelucrării efective) şi stocarea de lungă durată imaginilor, în fişiere, pe suporturi externe de memorie (benzi, discuri, etc.). Diferenţa esenţială între cele două tipuri de stocare este aceea că în memorie imaginea va fi reprezentată complet, în formă necomprimată, pentru a permite accesul rapid direct la informaţia fiecărui pixel.

1.3.1 Stocarea imaginilor în memorie

Principalul limbaj de programare utilizat pentru aplicaţii cu calcule intensive rămâne încă limbajul C (C++). Stocarea imaginilor se va face, evident, prin intermediul unor variabile ce implementează structuri de date bidimensionale. Ceea ce este deosebit este modul de declarare a acestora: declararea statică nu este convenabilă din cauza dimensiunilor în general mari ale imaginilor, şi deci este necesară o declarare dinamică. Particularitatea este dată de memorarea separată a fiecărei linii (sau coloane) a matricii într-un vector alocat dinamic, şi gruparea adreselor de început a acestora într-un vector de pointeri, la care se va reţine adresa de început (deci un alt pointer). Dacă considerăm un tip generic de date pentru componentele matricii (caracter, sau întreg, sau real), atunci o secvenţă C de declarare a unui imagini poate fi:

```
tip ** imagine;
unsigned int contor;
imagine=(tip**) malloc(nr_linii*sizeof(tip*));
for (contor=0;contor<nr_linii;contor++)
  imagine[contor]=(tip*) malloc(nr_coloane*sizeof(tip));
```

Se remarcă folosirea constantelor *nr_linii* şi *nr_coloane* (cu semnificaţie evidentă) şi a tipului generic *tip* pentru valoarea pixelilor. Linia a 3-a alocă spaţiul pentru un masiv de pointeri la date de tip pointer; spaţiul de memorie necesar (argumentul funcţiei *malloc*) este dat de numărul de pointeri la liniile imaginii ce înmulţeşte dimensiunea unui astfel de pointer (*sizeof(tip*)*). Valoarea *imagine[contor]* este adresa de început a spaţiului de memorie la care încep valorile pixelilor de pe linia *contor*; aceştia sunt stocaţi într-un vector declarat de *malloc(nr_coloane*sizeof(tip))*. Trebuie remarcată conversia de tip (cast) obligatorie ce însoţeşte fiecare alocare de memorie (se ştie că funcţia *malloc* întoarce un pointer la *void*). De asemenea se observă că secvenţa anterioară nu face nici un fel de verificare a succesului operaţiei de alocare de memorie (verificarea faptului că valoarea returnată de funcţia *malloc* nu este *NULL*). În mod implicit, la compilare, toţi pixelii (toate valorile matricii *imagine*) sunt iniţializaţi cu 0.

Spre deosebire de C, limbajul Matlab[3] aduce mari simplificări. Există un singur tip de date, reprezentate pe 8 octeţi (caracteristică ce se schimbă începând cu versiunea 5.0, ce admite valori reale, întregi sau caracter, declarate explicit). Orice variabilă Matlab este creată în momentul folosirii sale în membrul stâng al unei expresii (deci nu este necesară declararea prealabilă folosirii); orice variabilă este o matrice (scalarul este o matrice de o linie şi o coloană). Funcţiile returnează matrici. Secvenţa C anterioară este echivalentă cu:

```
imagine=zeros(nr_linii,nr_coloane);
```

[3] Codurile Matlab prezentate în lucrare corespund versiunii Matlab 4.2c.

1.3.2 Stocarea imaginilor în fişiere

Un fişier este entitatea logică de organizare a informaţiei înscrise pe mediile magnetice de stocare şi se compune dintr-un şir de octeţi. Pentru stocarea imaginii este necesar ca aceşti octeţi să conţină informaţia aferentă pixelilor precum şi informaţie despre tipul imaginii: dimensiunile acesteia, dacă este sau nu indexată, dacă are sau nu o tabelă de culoare ataşată, dacă este sau nu comprimată şi după ce metodă. Anumite structuri de fişiere au fost impuse de-a lungul timpului de firme producătoare de software sau de organisme de standardizare, căpătând denumirea de formate de imagini. Formatele de imagini s-au făcut cunoscute mai ales după extensia standard a fişierelor ce conţin imaginile stocate după formatul respectiv: BMP, TIF, GIF, PCX, JPG ... . În cele ce urmează ne vom referi la formatele RAW(cunoscut şi ca IMG), unul dintre cele mai rudimentare formate de fişiere imagine, şi Windows Bitmap -BMP al firmei Microsoft, care este unul dintre cele mai larg recunoscute formate de fişiere.

Un fişier RAW conţine imagini indexate cu nivele de gri, de formă pătrată. Fişierul nu are antet (dimensiunile imaginii fiind deduse din dimensiunea fişierului ce o conţine) şi nu conţine nici tabel de culoare (acesta are toate componentele liniei i egale între ele, reprezentând griuri). Fiecare pixel al imaginii este codat cu numărul corespunzător de biţi (4, 8, etc.); imaginea este baleiată în ordinea normală (începând cu prima linie a imaginii, de la stânga la dreapta).

Un fişier BMP[4] are trei componente consecutive: un antet de fişier (BITMAPFILEHEADER), o structură de informaţie a imaginii(BITMAPINFO) şi codarea pixelilor. Antetul de fişier (BITMAPFILEHEADER) conţine informaţii asupra tipului, dimensiunii şi reprezentării fişierului Bitmap independent de dispozitiv (DIB - Device Independent Bitmap); semnificaţiile componentelor sunt date în tabelul 1.4.

```
typedef struct tagBITMAPFILEHEADER{
WORD bfType;
DWORD bfType;
WORD bfReserved1;
WORD bfReserved2;
DW bfOffBits;
}BITMAPFILEHEADER;
```

Structura de informaţie a imaginii (BITMAPINFO) conţine informaţii asupra dimensiunilor şi culorilor unui DIB, şi este alcătuită din două componente: antetul structurii de informaţii (BITMAPINFOHEADER), a cărui componente sunt descrise în tabelul 1.5 şi tabelul de culoare, format din structuri RGBQUAD.

[4]Denumirile componentelor logice ale fişierului sunt cele standardizate de Microsoft.

Câmp	Descriere
bfType	Specifică tipul de fişier; trebuie să conţină caracterele BM
bfSize	Specifică lungimea fişierului în DWORD
bfReserved1	Câmp rezervat, valoare 0
bfReserved2	Câmp rezervat, valoare 0
bfOffBits	Specifică deplasamentul în octeţi de la sfârştul structurii BITMAPFILEHEADER până la zona din fişier ce conţine pixelii codaţi

Tabel 1.4: Descrierea câmpurilor structurii BITMAPFILEHEADER

Câmp	Descriere
biSize	Numărul de octeţi ai structurii BITMAPINFOHEADER
biWidth	Lăţimea imaginii, în pixeli
biHeight	Înălţimea imaginii, în pixeli
biPlanes	Numărul de plane de culoare ale dispozitivului de afişaj (1)
biBitCount	Numărul de biţi cu care se codează un pixel; poate fi 1, 4, 8 sau 24
biCompression	Tipul de compresie utilizată: BI_RGB fără compresie, BI_RLE8 sau BI_RLE4 pentru compresie de tip RLE cu cuvinte de respectiv 8 sau 4 biţi
biSizeImage	Dimensiunea imaginii în octeţi
biXPelsPerMeter	Rezoluţia pe orizontală a dispozitivului ţintă (în pixeli pe metru)
biYPelsPerMeter	Rezoluţia pe verticală a dispozitivului ţintă (în pixeli pe metru)
biClrUsed	Numărul de culori utilizate în imagine; dacă este 0, imaginea foloseşte toate culorile disponibile ale paletei
biClrImportant	Numărul de culori considerate importante; dacă este 0, toate culorile sunt luate în considerare

Tabel 1.5: Descrierea câmpurilor structurii BITMAPINFOHEADER

```
typedef struct tagBITMAPINFOHEADER{
DWORD biSize;
DWORD biWidth;
DWORD biHeight;
WORD biPlanes;
WORD biBitCount;
DWORD biCompression;
DWORD biSizeImage;
DWORD biXPelsPerMeter;
DWORD biYPelsPerMeter;
DWORD biClrUsed;
DWORD biClrImportant;
} BITMAPINFOHEADER;
```

Structura RGBQUAD descrie o culoare prin componentele sale de roşu, verde şi albastru, şi un câmp rezervat având valoarea 0.

```
typedef struct tagRGBQUAD{
BYTE rgbBlue;
BYTE rgbGreen;
BYTE rgbRed;
BYTE rgbReserved;
}RGBQUAD;
```

Codarea pixelilor se face după câteva reguli. Fiecare pixel va fi codat pe biBitCount biţi; dacă biBitCount este 1, 4 sau 8, imaginea va fi indexată şi fişierul conţine tabela de culoare asociată imaginii. Codurile pixelilor se grupează pe octeţi (deci pentru o codare de 4 biţi per pixel, fiecare octet de cod va corespunde la doi pixeli alăturaţi). Dacă biBitCount este 24, pentru fiecare pixel se asociază direct trei octeţi, ce reprezintă componentele de roşu, verde şi albastru ale culorii respective; această imagine se numeşte *True Color* şi nu mai are un tabel de culoare asociat. Denumirea de *True Color* (culoare adevărată) provine din faptul că numărul total de culori ce se pot astfel reprezenta (2^{24}) depăşeşte limita sensibilităţii umane de discernere a culorilor.

Codarea se face independent pe fiecare linie orizontală a imaginii. Codurile (indexurile) tuturor pixelilor unei linii sunt concatenate; şirul rezultat trebuie să fie multiplu de 32 de biţi (sau de 4 octeţi, sau să conţină un număr întreg de DWORD's). Dacă acestă constrângere nu este respectată, linia respectivă se completează cu numărul necesar de biţi (care, în mod evident, nu vor fi utilizaţi la citirea imaginii din fişier). Codarea imaginii începe cu ultima linie, şi pentru fiecare linie baleiajul este normal (de la stânga la dreapta).

Capitolul 2

TEHNICI DE ÎMBUNĂTĂŢIRE A IMAGINILOR

Îmbunătăţirea imaginilor este o sintagmă generală ce se referă la o clasă largă de operaţii al căror scop este mărirea detectabilităţii componentelor imaginii. Detectabilitatea componentelor este legată mai mult de percepţia vizuală a unui observator uman decât de o analiză automată cantitativă. Percepţia vizuală de referinţă este cea a unui expert uman în domeniul aplicaţiei din care provine imaginea.

Aşadar criteriile de evaluare ale calităţii unei imagini sunt subiective şi specifice aplicaţiei. În [2] se face analogia operaţiilor de îmbunătăţire a imaginilor cu reglajul tonalităţii muzicii ascultate; în funcţie de ascultător, se vor favoriza componentele înalte sau joase, sau nici unele. Ca o consecinţă, procesul de îmbunătăţire va fi interactiv, transformările efectuate trebuind să fie validate (cel puţin în etapa de proiectare sau probă) de către un utilizator uman.

Principiul (aproape unanim acceptat) este că îmbunătăţirea calităţii unei imagini se face fără a lua în considerare nici o informaţie asupra imaginii originale sau asupra procesului de degradare (prin care imaginea nu este suficient de bună). Conform acestui punct de vedere chiar şi o imagine originală (nedegradată) poate fi îmbunătăţită, obţinând o imagine falsificată, dar subiectiv preferabilă [18]. În general, calitatea subiectivă a unei imagini poate fi apreciată pe baza contrastului sau accentuării elementelor de contur (muchii, frontiere, linii, margini) şi pe baza netezimii în regiunile uniforme.

Creşterea uniformităţii regiunilor este însă asimilată eliminării unui eventual zgomot suprapus imaginii, operaţie denumită în mod clasic filtrare. Filtrarea ce are ca scop eliminarea zgomotului va fi studiată în următoarele capitole.

Din punctul de vedere al metodelor utilizate, putem distinge mai multe tipuri de operaţii de îmbunătăţire:

- operaţii punctuale, prin care se realizează o corespondenţă de tip "unu la unu" între vechea valoare a nivelului de gri şi noua valoare a acestuia, pentru fiecare pixel al imaginii. Tot în acestă categorie vom include şi operaţiile de pseudocolorare, care se referă la afişarea imaginii folosind o paletă de culoare modificată.
- operaţii locale (sau de vecinătate), prin care noua valoare a nivelului de gri într-un pixel este obţinută din vechea valoare a pixelului repectiv şi din valorile unor pixeli vecini pixelului considerat.
- operaţii integrale, în care noua valoare a unui pixel este dependentă de valorile tuturor pixelilor imaginii

În acest capitol vom studia doar operaţiile punctuale şi de pseudocolorare.

Prin îmbunătăţire, unei imagini nu i se adaugă nici o informaţie nouă faţă de cea ce exista iniţial [9] (deci nu se adaugă nimic imaginii), ci doar este prezentat altfel conţinutul iniţial al acesteia. Deşi la o examinare superficială afirmaţia este corectă, putem găsi măcar două obiecţii (sau contraexemple) la această formulare:

- din punctul de vedere al utilizatorului, informaţia, chiar dacă există, nu poate fi folosită, deci este asimilabil nulă. Acesta este cazul imaginilor obţinute în condiţii extreme de iluminare, ce prezintă un contrast foarte slab (imagini subexpuse sau supraexpuse) [5].
- din punctul de vedere al teoriei informaţiei, informaţia din imagine poate fi asimilată entropiei procesului aleator ale cărui realizări particulare sunt valorile de gri ale pixelilor. Entropia se modifică însă la orice transformare ce afectează distribuţia nivelelor de gri din imagine.

2.1 Operaţii punctuale de modificare a contrastului

Operaţiile punctuale de modificare a contrastului (numite şi transformări ale nivelului de gri) sunt asocieri (*mapping*, în engleză) ce leagă nivelul de gri original de noua sa valoare. O asemenea asociere nu este altceva decât o funcţie:

$$v = T(u),\ u \in [0; L-1] \tag{2.1}$$

În [5] se stabilesc ca necesare condiţiile ca:

- transformarea T să păstreze gama admisibilă de valori ale imaginii (dacă nivelele de gri au fost reprezentate pe L nivele de cuantizare, atunci $0 \leqslant T(u) \leqslant L-1$, $\forall u \in [0; L-1]$)

- transformarea T să fie monotonă (crescătoare sau descrescătoare) pentru a păstra ordinea între nivelele de gri

2.1.1 Modificarea liniară a contrastului

Cea mai des folosită tehnică de modificare liniară a contrastului este o transformare liniară pe porţiuni, dată de:

$$v = \begin{cases} \frac{\alpha}{T_1} u, \, 0 \leq u < T_1 \\ \alpha + \frac{\beta - \alpha}{T_2 - T_1} \left(u - T_1 \right), \, T_1 \leq u < T_2 \\ \beta + \frac{L-1-\beta}{L-1-T_2} \left(u - T_2 \right), \, T_2 \leq u < L \end{cases} \tag{2.2}$$

În formula anterioară, parametrii de control sunt T_1, T_2, α şi β; aceştia sunt grupaţi câte doi, definind punctele (T_1,α) şi (T_2,β). Aceste două puncte de control, împreună cu punctele fixe $(0, 0)$ şi $(L-1, L-1)$ vor defini cele trei segmente de dreaptă ce apar în formula (2.2). Rezultatul aplicării unei asemenea operaţii punctuale se obţine modificând valoarea (nivelul de gri) fiecărui pixel al imaginii iniţiale, u, conform (2.2), obţinând noul nivel de gri v. Transformarea poate fi făcută în două moduri: fie se repetă calculele de la (2.2) pentru fiecare pixel, baleind imaginea, fie noile valori ale contrastului se calculează de la început pentru toate nivelele de gri posibile (între 0 şi $L-1$) şi apoi aceste modificări se aplică imaginii. Codul C următor implementează a doua variantă de calcul, care este mai rapidă (calculele nivelelor de gri au fost separate de ciclul de baleiere al imaginii şi au fost eliminate structurile condiţionale *if* - impuse de definiţia de tip "acoladă" - prin separarea domeniilor de calcul în trei cicluri). Trebuie remarcată de asemenea definirea nivelului de gri ca *unsigned int*, ceea ce crează posibilitatea folosirii unui număr de nivele de gri mai mare de 256 (pentru care ar fi fost suficient un *unsigned char*).

```
unsigned int T1,T2,alfa,beta,gri_vechi,i,j;
gri_nou=(unsigned int *) malloc (L*sizeof(unsigned int));
for (gri_vechi=0;gri_vechi<T1;gri_vechi++)
  gri_nou[gri_vechi]=alfa*gri_vechi/T1;
for (gri_vechi=T1;gri_vechi<T2;gri_vechi++)
  gri_nou[gri_vechi]=alfa+(beta-alfa)*(gri_vechi-T2)/(T2-T1);
for (gri_vechi=T2;gri_vechi<L;gri_vechi++)
  gri_nou[gri_vechi]=beta+(L-1-beta)*(gri_vechi-T1)/(L-1-T2);
for (i=0;i<NRLIN;i++)
  for (j=0;j<NRCOL;j++)
    imagine_noua[i][j]=gri_nou[imagine_veche[i][j]];
```

Vom prezenta în cele ce urmează un exemplu; în figura 2.1 este prezentată pe de o parte imaginea originală "lena" (una dintre fotografiile impuse ca standard de fapt în raportarea rezultatelor obţinute), iar alăturat imaginea obţinută cu o modificare liniară pe porţiuni

a contrastului, determinată de parametrii $T_1 = 30$, $T_2 = 100$, $\alpha = 20$, $\beta = 200$. În figura 2.2 este prezentată funcţia de transformare a nivelelor de gri ($T(u)$).

Fig. 2.1: Imagine originală şi imagine cu contrastul modificat

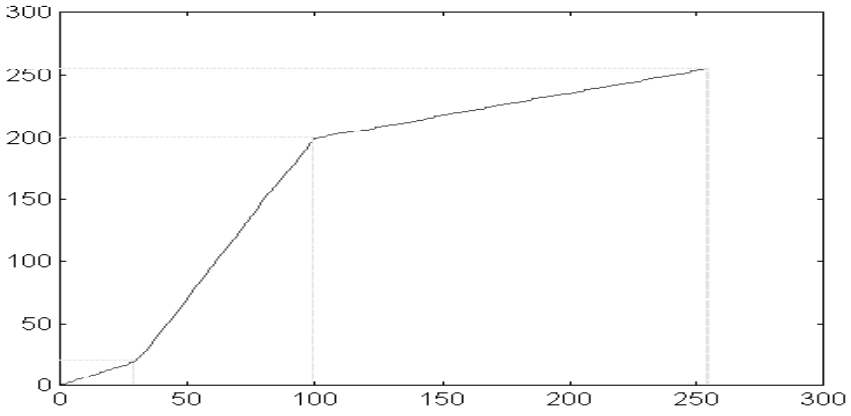

Fig. 2.2: Îmbunătăţire liniară pe porţiuni, definită de punctele (0,0), (30,20), (100,200), (255,255)

Vizibilitatea componentelor scenei este în general determinată în cea mai mare parte de contrastul zonei din imagine; contrastul este o măsură proporţională cu diferenţa dintre luminozitatea anumitor pixeli (nivelul lor de gri). Pentru a putea prevedea deci efectele unei operaţii de îmbunătăţire de tipul prezentat asupra contrastului este deci suficientă studierea diferenţelor de nivele de gri între o aceeaşi pereche de pixeli înainte şi după efectuarea transformării. La limită, este posibil ca pixelii să aibă nivelul de gri original diferit cu doar o unitate (cuanta minimă), şi atunci modificarea contrastului va fi dată

de diferenţa valorilor transformate, adică de derivata funcţiei de transformare:

$$C = \frac{\Delta v}{\Delta u} = \frac{T(u_2) - T(u_1)}{u_2 - u_1} = \frac{dT(u)}{du} = T'(u) \tag{2.3}$$

Pentru funcţia liniară pe porţiuni este evident că derivata va fi constantă pe aceleaşi intervale, având valoarea egală cu panta segmentului de dreaptă.

$$C = \begin{cases} \frac{\alpha}{T_1}, \text{ dacă } u \in [0; T_1] \\ \frac{\beta-\alpha}{T_2-T_1}, \text{dacă } u \in [T_1; T_2] \\ \frac{L-1-\beta}{L-1-T_2}, \text{ dacă } u \in [T_2; L-1] \end{cases} \tag{2.4}$$

Dacă pe un interval această pantă este subunitară, atunci diferenţa între nivelele alăturate de gri se micşorează şi deci contrastul scade; dacă din contră, panta dreptei este supraunitară, diferenţa dintre nivelele de gri alăturate se măreşte şi contrastul va creşte. Spre exemplu, în transformarea din figura 2.2, pe intervalul [30,100] contrastul creşte, iar pe intervalele [0,30] şi [100,255] contrastul scade. Aceste efecte sunt uşor vizibile pe imaginile din figura 2.1: de exemplu scăderea contrastului pentru nivelele de gri de la capătul superior al gamei admise (dinspre alb) se observă prin dispariţia detaliilor luminoase din imagine (panglica lată de la pălărie); creşterea contrastului pe intervalul [30,100] (deci griuri închise) este vizibil în zona penei de pălărie, ale cărei detalii sunt acum mult mai sesizabile.

În funcţie de alegerea celor patru parametri, se pot obţine câteva cazuri particulare de interes ce poartă denumiri specifice.

Fig. 2.3: Imagine originală şi imagine binarizată cu pragul 125

Dacă $T1 = T2$ şi $\alpha = 0$, $\beta = L - 1$, se obţine prăguirea sau binarizarea ("thresholding") (vezi figura 2.4); în imaginea rezultată nu există decât alb şi negru (figura 2.3); toate

nivelele de gri iniţiale a căror valoare era mai mică decât T_1 fiind negre şi toate nivelele de gri iniţiale mai mari ca T_1 devenind albe. După cum se va vedea la capitolul de segmentare orientată pe regiuni (capitolul 8), aceasta este şi una dintre tehnicile cele mai simple de segmentare. În urma acestei transformări, contrastul este maximizat la nivelul întregii imagini.

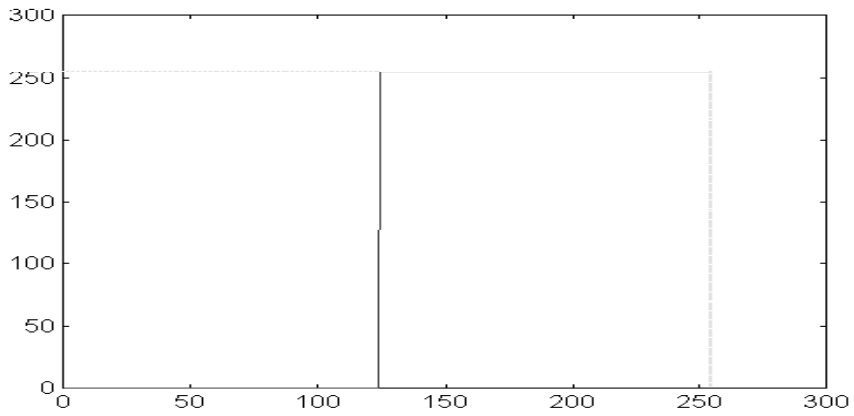

Fig. 2.4: Transformarea de binarizare

Dacă $\alpha = 0$ şi $\beta = L - 1$ se obţine operaţia de întindere maximă a contrastului (*contrast streching*) (vezi figura 2.5) pentru intervalul $[T_1; T_2]$. Nivelele de gri care se găsesc în afara acestui interval vor fi înlocuite fie cu alb, fie cu negru.

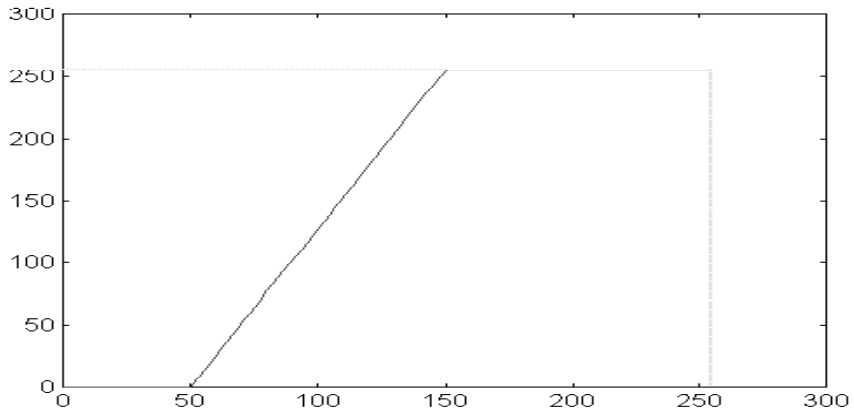

Fig. 2.5: Transformarea de întindere maximă a contrastului

2.1.2 Modificarea neliniară a contrastului

Principalul dezavantaj al tehnicii liniare pe porţiuni prezentate este faptul că modificarea contrastului este aceeaşi pe un întreg interval de nivele de gri, şi nu este posibilă o modificare neuniformă a contrastului pe întregul interval de nivele de gri sau în jurul unui anume nivel de gri. Tehnicile neliniare au aceste proprietăţi.

O primă variantă este compandarea domeniului [9], definită de o curbă logaritmică şi cu punctele fixe $(0,0)$ şi $(L-1, L-1)$:

$$v = T(u) = \frac{L-1}{\lg L} \lg(1+u) \tag{2.5}$$

Contrastul va varia neuniform de-a lungul scalei de gri, mărindu-se la capătul inferior (negru) şi micşorându-se la capătul superior (alb). În mod reciproc se poate defini expandarea domeniului, ca transformare inversă celei de compandare, şi deci având o alură exponenţială:

$$v = T(u) = (L-1)\frac{e^u - 1}{e^{L-1} - 1} \tag{2.6}$$

Contrastul va varia neuniform de-a lungul scalei de gri, mărindu-se la capătul superior (alb) şi micşorându-se la capătul inferior (negru). Termenii de compandare şi de expandare au fost daţi prin asemănare cu transformările folosite în teoria codării şi cuantizării (ce intervin în metodele de cuantizare a semnalului vocal pentru telefonia digitală, cunoscute sub numele de legea A în Europa şi legea μ în America). Trebuie însă subliniat că în prelucrarea imaginilor aceste transformări nu afectează domeniul de valori, care rămâne $[0, L-1]$.

Alte transformări neliniare pot fi obţinute prin folosirea unor funcţii de tip putere; şi acestea au nivelele de gri extreme ca puncte fixe ($(0,0)$ şi $(L-1, L-1)$). O primă variantă este funcţia putere:

$$v = T(u) = (L-1)\left(\frac{u}{L-1}\right)^r \tag{2.7}$$

După valorile parametrului-putere r se pot obţine două comportări diferite: pentru $r < 1$ comportarea este de acelaşi tip cu al funcţiei de compandare logaritmice, iar pentru $r > 1$ comportarea este de tipul funcţiei de expandare. Trebuie remarcat că legile de variaţie ale contrastului vor fi însă diferite.

Există însă şi o variantă la care se mai adaugă un punct fix (T, T), funcţia devenind cu două intervale de definiţie:

$$v = T(u) = \begin{cases} T\left(\frac{u}{T}\right)^r, \text{ dacă } u \in [0;T] \\ L-1-(L-1-T)\left(\frac{L-1-u}{L-1-T}\right)^r, \text{ dacă u} \in [T, L-1] \end{cases} \tag{2.8}$$

Funcţia are o alură de tipul celei prezentate în figura 2.6.

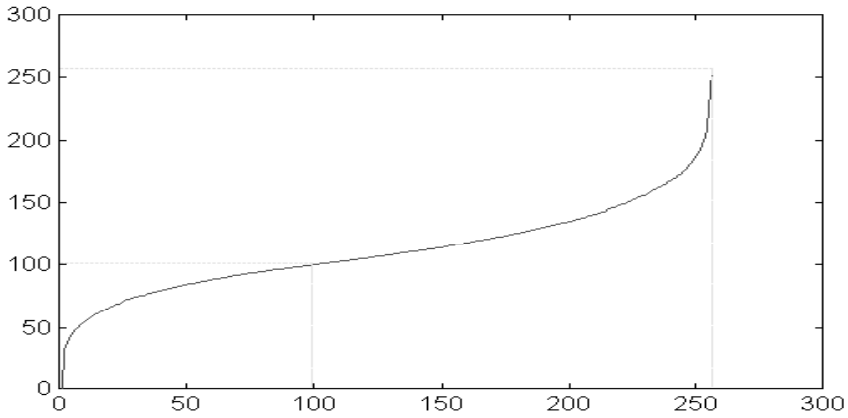

Fig. 2.6: Modificare neliniară a contrastului, cu trei puncte fixe

În [9], în cadrul operaţiilor punctuale de îmbunătăţire a imaginilor sunt prezentate şi operaţii aritmetice simple, ca de exemplu negativarea. Negativarea este descrisă de:

$$v = T(u) = L - 1 - u \tag{2.9}$$

Efectul acesteia de modificare a contrastului se bazează doar pe caracteristicile sistemului vizual uman, pentru care contrastul depinde de diferenţa de luminozitate între pixeli aparţinând unui obiect, respectiv fundalului, raportată la luminanţa medie a fundalului. O categorie aparte de aplicaţii în care sunt utile asemenea inversiuni este analiza imaginilor medicale, care pentru o analiză automată trebuiesc inversate; un astfel de exemplu este imaginea angiografică din figura 2.7.

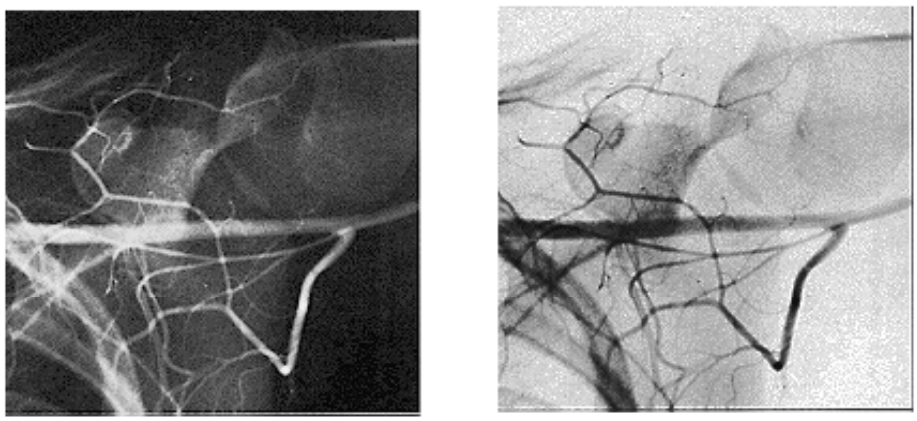

Fig. 2.7: Imagine originală şi negativată (dintr-o aplicaţie medicală)

Alte operaţii posibile (tratate în [9], dar de mai mică semnificaţie practică) sunt repre-

zentarea planelor de bit (pentru fiecare pixel al imaginii, fiecare bit al reprezentării binare a nivelului de gri este considerat ca valoarea unui pixel al unei imagini binare), transformarea de lipire *clipping* (care păstrează nemodificate nivelele de gri dintr-un anumit interval şi le anulează pe celelalte - imaginea rezultată conţinând obiectele de interes pe fond negru) şi transformarea de tăiere *slicing* (care transformă nivelele de gri dintr-un interval fixat în alb şi tot restul în negru; nu este altceva decât un alt tip de binarizare).

2.2 Pseudocolorarea

Pseudocolorarea este o tehnică de îmbunătăţire a vizibilităţii anumitor componente ale imaginii (sau a imaginii în ansamblu) prin modificarea paletei de culoare cu care imaginea este afişată (reprezentată). Aceasta înseamnă că pentru anumite nivele de gri, afişarea nu se va mai face cu culoarea a cărei componente sunt toate egale cu indexul (nivelul de gri), ci cu o altă culoare. Acestă definiţie acoperă însă şi cazul operaţiilor de modificare a contrastului prezentate anterior; funcţia $v = T(u)$ nu este altceva decât o funcţie de construcţie a unei noi palete de culoare pentru aceeaşi imagine. Spre exemplu, codul modificării neliniare de contrast devine:

```
unsigned int T1,T2,alfa,beta,gri_vechi;
gri_nou=(unsigned int *) malloc (L*sizeof(unsigned int));
for (gri_vechi=0;gri_vechi<T1;gri_vechi++)
  gri_nou[gri_vechi]=alfa*gri_vechi/T1;
for (gri_vechi=T1;gri_vechi<T2;gri_vechi++)
  gri_nou[gri_vechi]=alfa+(beta-alfa)*(gri_vechi-T2)/(T2-T1);
for (gri_vechi=T2;gri_vechi<L;gri_vechi++)
  gri_nou[gri_vechi]=beta+(L-1-beta)*(gri_vechi-T1)/(L-1-T2);
for (gri_vechi=0;gri_vechi<L;gri_vechi++)
  setpalette(gri_vechi,(color) gri_nou[gri_vechi]);
```

Ultima buclă *for* a codului modifică paleta de culoare pentru fiecare index (intrare) a acesteia, conform noilor valori calculate. Remarcaţi cast-ul (schimbarea de tip) la tipul *color*; acesta este inserat mai mult ca o măsură de atenţionare: culoarea (deci variabila de tip *color)* trebuie obţinută din scalarul nivel de gri în conformitate cu regulile de descriere a culorilor valabile în respectivul mod grafic.

Ideea de bază în pseudocolorare este de a folosi culori pentru a pune în evidenţă zone de interes din imagini cu nivele de gri (există şi varianta colorării false - *false coloring*, care transformă o imagine color într-o altă imagine color, dar cu un contrast mult mai pronunţat şi artificial între elementele sale). Acestă idee este normală dacă se are în vedere faptul că ochiul (sistemul vizual) uman distinge ceva mai puţin de 256 nuanţe de gri, deşi diferenţiază câteva milioane de culori [9].

O aplicaţie interesantă a pseudocolorării este prezentată în [2]. La NASA, la începuturile erei digitale în tehnicile de achiziţie a imaginilor, era necesară digitizarea unor imagini de microscopie, pentru care iluminarea era reglată manual, până la o vizibilitate maximă a tuturor detaliilor. Pentru că operatorul nu putea distinge clar şi precis care era iluminarea cea mai potrivită (care nici nu producea suprailuminare şi nici nu lăsa umbre mai mari decât era necesar), la afişarea în timp real a imaginii achiziţionate pe monitoarele de control, paleta de griuri a fost modificată pe ultima poziţie (alb), unde s-a inserat roşu. Atunci instrucţiunile pentru operator erau: creşte curentul prin lampă (iluminarea) până când în imagine apare roşu, după care redu curentul până când culoarea roşie dispare. Rezultatul acestei tehnici simple au fost mii de imagini digitizate corect.

În final merită poate amintită remarca (destul de acidă) din [2]:

> "Deşi prin natura sa este un detaliu al tehnicilor de afişare, pseudocolorarea a fost adesea glorificată prin termeni ca prelucrare prin pseudocolorare sau analiză prin pseudocolorare. Pseudocolorarea rămâne un accesoriu favorit al vânzătorilor, care o utilizează adesea în demonstraţiile produselor [software], deoarece poate stârni interesul în ochii clienţilor mult mai repede decât orice altă metodă de afişare cunoscută. Cercetările mele au adus la lumină o listă dureros de scurtă a aplicaţiilor demonstabil productive a pseudocolorării"

2.3 Operaţii de contrastare bazate pe histograma imaginii

Pentru o imagine f de $M \times N$ pixeli şi L nivele de gri, histograma este definită (2.10) ca probabilitatea (frecvenţă relativă) de apariţie în imagine a diferitelor nivele de gri posibile.

$$h(i) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \delta(i - f(m,n)) \quad , \quad i = 0, 1, ...L-1 \tag{2.10}$$

Din punct de vedere statistic, putem considera valoarea fiecărui pixel al imaginii ca o realizare particulară a unei variabile aleatoare asociată nivelelor de gri, caz în care histograma (2.10) este funcţia de densitate de probabilitate a acestei variabile aleatoare. Fiind o funcţie de densitate de probabilitate, histograma oricărei imagini verifică condiţia de normare $\sum_{i=0}^{L-1} h(i) = 1$.

Din punct de vedere practic, calculul histogramei unei imagini înseamnă parcurgerea punct cu punct a imaginii şi contorizarea numărului de nivele de gri întâlnite. Presupunând L nivele de gri posibile în imaginea de dimensiuni $NRLIN$ şi $NRCOL$, codul

următor alocă pentru fiecare nivel de gri posibil câte un *unsigned int*, ce va contoriza numărul de apariţii ale nivelului de gri respectiv. Pentru fiecare punct al imaginii se incrementează poziţia din histogramă ce corespunde valorii de gri din acel pixel. Ceea ce rezultă în final diferă de histograma descrisă de (2.10) prin constanta de normare "numărul total de puncte ale imaginii" (deci MN sau *NRLIN*NRCOL*); este evident că valorile obţinute pot fi normate pentru a obţine o funcţie de densitate de probabilitate prin împărţirea cu *NRLIN*NRCOL* şi definirea histogramei ca fiind formată din *real* sau *float*.

```
histo=(unsigned int*)malloc(L*sizeof (unsigned int));
for (i=0;i<L;i++)
  histo[i]=0;
for (i=0;i<NRLIN;i++)
  for (j=0;j<NRCOL;j++)
    histo[imagine[i][j]]++;
```

În Matlab, implementarea oricărei funcţii este cu atât mai eficientă (din punctul de vedere al timpului de rulare) cu cât sunt evitate structurile repetitive (în particular buclele *for*). Cum, pentru calculul histogramei, ciclarea nu poate fi evitată (deci trebuie parcurse fie toate punctele imaginii, fie toate nivelele de gri), se alege varianta care implică repetarea minimă a ciclării:

```
histo=zeros(1,L);
for i=1:L
  p=find(imagine==i);
  histo(i)=length(p);
% histo(i)=length(p)/prod(size(imagine));
end
```

Funcţia *find* (funcţie standard a pachetului Matlab) returnează poziţiile (indicii) la care este găsită valoarea i în matricea *imagine* (adică întoarce poziţiile punctelor ce au valoarea i); numărând aceste puncte (deci calculând lungimea vectorului în care sunt stocate) se găseşte numărul de puncte ce au respectivul nivel de gri. Normarea histogramei se poate face fără a modifica declaraţiile de definire a structurilor (pentru Matlab este indiferent dacă se stochează întregi sau numere cu parte zecimală). Acestă structură este mai rapidă decât cea prin care se parcurgea toată imaginea deoarece se foloseşte o singură buclă de lungime L (şi nu două bucle imbricate, de lungime totală *NRLIN*NRCOL),* iar funcţia *find* este rapidă, fiind o funcţie precompilată.

Histograma imaginii oferă informaţii asupra plasamentului în "nuanţă" a conţinutului imaginii (vezi figura 2.8). La majoritatea imaginilor există o distribuţie neuniformă a nivelelor de gri; există nivele de gri predominante şi există nivele de gri folosite puţin sau

deloc. Operaţiile de îmbunătăţire a imaginilor (pentru îmbunătăţirea percepţiei vizuale) au ca scop redistribuirea nivelelor de gri, astfel ca acestea să ocupe întreaga gamă de variaţie disponibilă, în mod uniform: aceasta este egalizarea de histogramă [9], [18].

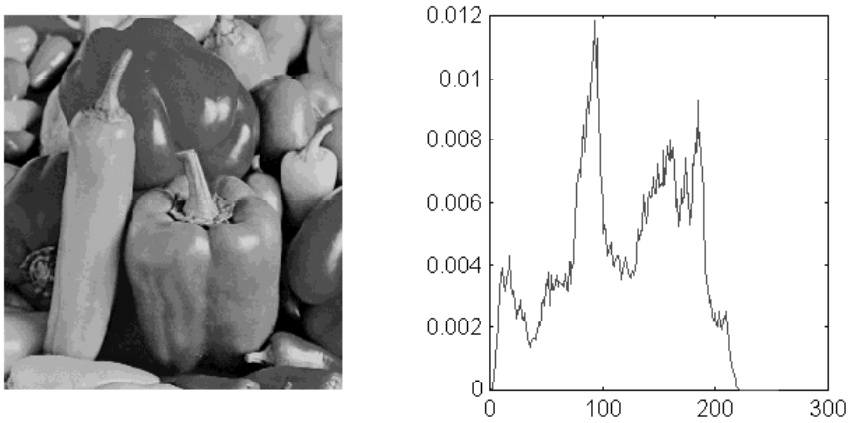

Fig. 2.8: Imagine cu nivele de gri şi histograma acesteia

Scopul egalizării de histogramă este deci obţinerea unei distribuţii uniforme a nivelelor de gri; imaginea rezultată va prezenta cea mai mare îmbunătăţire a contrastului, distribuit regulat în întreaga gamă dinamică a nivelelor de gri. Din punct de vedere matematic, egalizarea de histogramă înseamnă transformarea unei distribuţii oarecari (descrisă de histograma imaginii iniţiale) într-o distribuţie uniformă. Considerând variabilele aleatoare $X(\xi, x)$ şi $Y(\xi, y)$ legate prin $Y = g(X)$, atunci între funcţiile de densitate de probabilitate a celor două variabile aleatoare există relaţia [16]:

$$f_Y(y) = f_X(x) \frac{1}{\left|(g^{-1}(y))'\right|}|_{x=g^{-1}(y)}$$

Dacă dorim ca funcţia de densitate de probabilitate $f_Y(y)$ să fie uniformă (în condiţiile în care funcţia de densitate de probabilitate $f_X(x)$ este dată), atunci înseamnă că vom avea $\left|(g^{-1}(y))'\right| = \frac{1}{k} f_X\left(g^{-1}(y)\right)$. Rezolvarea acestei ecuaţii produce soluţia $y = g(x) = \int_{-\infty}^{x} f_X(t)dt$.

Pentru cazul particular al imaginilor, variabila aleatoare X ia valori naturale - nivelele de gri. Funcţia de densitate de probabilitate $f_X(x)$ este histograma [normată] a imaginii iar funcţia de transformare devine $y = g(x) = \int_{0}^{x} f_X(t)dt$. Ţinând seama că valorile de gri sunt discrete, integrala se transformă în sumă şi acestă formă nu este altceva decât

histograma cumulativă a imaginii; dacă h este histograma imaginii, atunci

$$g(x) = H(x) = \sum_{i=0}^{x} h(i) \tag{2.11}$$

Valorile funcţiei trebuie însă redistribuite în intervalul permis de valori de gri, ceea ce duce la deducerea formulei care exprimă noile valori de gri:

$$v = \left[\frac{H(u) - H(0)}{MN - H(0)} (L-1) + 0.5 \right] \tag{2.12}$$

O variantă de cod care realizează egalizarea de histogramă este prezentată în continuare. Trebuie remarcat că modul de calcul al histogramei cumulative este de tip iterativ, bazându-se pe formula $H(x) = H(x-1) + h(x)$, ce se poate deduce imediat din (2.11). Mai trebuie de asemenea remarcat că se foloseşte o histogramă nenormată.

```
gri_nou=(unsigned int *) malloc (L*sizeof(unsigned int));
histo_cum=(unsigned int *) malloc (L*sizeof(unsigned int));
tot=NRLIN*NRCOL;
histo_cum[0]=histo[0];
for (i=1;i<L;i++)
  histo_cum[i]=histo_cum[i-1]+histo[i];
for (i=0;i<L;i++)
  gri_nou[i]=round((histo_cum[i]-histo_cum[0])*(L-1)/(tot-histo_cum[0]));
```

Figurile următoare prezintă o imagine originală ("lena") şi rezultatul egalizării de histogramă, precum şi histogramele originale şi egalizate (figura 2.9). Ceea ce se remarcă cu uşurinţă este diferenţa dintre histograma obţinută şi histograma perfect uniformă dorită. Unul dintre efectele secundare notabile este introducerea de nivele de gri lipsă în histograma egalizată (aspectul "în pieptene" al acesteia).

Imagine originală

Imagine după egalizarea de histogramă

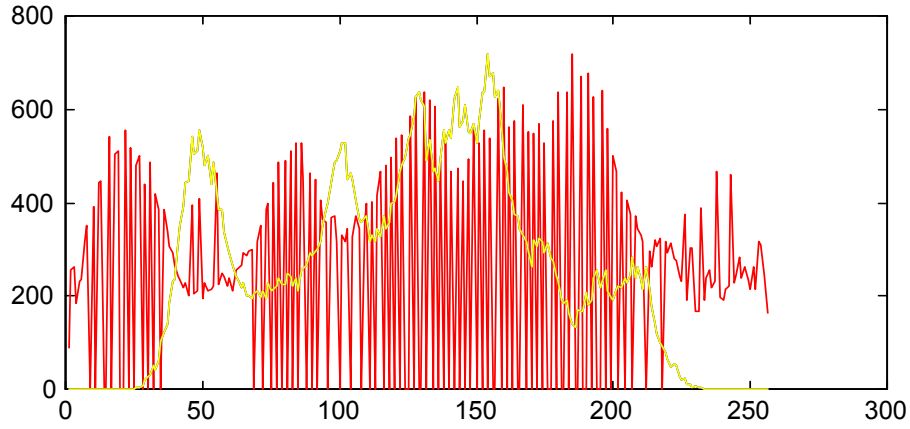

Fig. 2.9: Histograma imaginii "lena", înainte şi după egalizare

Alte variante de egalizare a histogramei [18] folosesc histograme modificate ale imaginii (prin contabilizarea doar a anumitor pixeli) sau limitează amplitudinea vârfurilor histogramei, redistribuind uniform pixelii în exces.

Tehnica de egalizare a histogramei poate fi extinsă prin realizarea unei histograme de formă impusă (dar oarecare) a imaginii rezultat; această metodă este denumită specificare de histogramă [9], [5].

O ultimă remarcă se poate referi la validitatea introducerii tehnicilor de îmbunătăţire a histogramei în categoria de operaţii punctuale. Majoritatea autorilor consideră că orice operaţie care este echivalentă cu modificarea paletei de culoare a imaginii este o operaţie punctuală. În acelaşi timp însă, noile valori de gri ale fiecărui punct sunt calculate pe baza histogramei imaginii, deci pe baza unei măsuri ce ia în calcul valorile din întreaga imagine; din acest punct de vedere, egalizarea de histogramă poate fi inclusă în categoria operaţiilor integrale.

Capitolul 3

FILTRAREA LINIARĂ A IMAGINILOR

Odată cu operaţiile de filtrare liniară, se începe studiul operatorilor de vecinătate: operatorii al căror rezultat depinde atât de valoarea punctului în care au fost aplicaţi, cât şi de valorile unui număr de alte puncte vecine (nu neapărat topologic) punctului curent de calcul. Filtrarea se cheamă liniară pentru că operaţia verifică principiul superpoziţiei (liniarităţii): pentru două imagini f_1 şi f_2, doi scalari a_1 şi a_2 şi operatorul liniar L avem

$$L(a_1 f_1 + a_2 f_2) = a_1 L(f_1) + a_2 L(f_2) \tag{3.1}$$

Operaţia de filtrarea liniară calculează noua valoare a unui pixel al imaginii (din poziţia (m, n)) ca o combinaţie liniară (medie ponderată) a unui număr de valori din imaginea originală:

$$v(m, n) = \sum_{(k,l) \in W} w_{kl} f(m - k, n - l) \tag{3.2}$$

W este vecinătatea punctului curent în care se face calculul; W este numită mască sau fereastră de filtrare şi este o formă plană, descrisă ca o mulţime de puncte din spaţiul cartezian, în coordonate relative (deci în alt sistem de coordonate decât sistemul de coordonate a imaginii). Scalarii w_{kl} sunt ataşaţi poziţiilor (k, l) din fereastra de filtrare şi poartă numele de coeficienţi ai filtrului. Spre exemplu, o mască simplă de mediere este media aritmetică a valorilor dintr-o vecinătate pătrată de 3 x 3 pixeli, centrată în pixelul curent; pentru acest caz, toţi coeficienţii vor avea valoarea $1/9$ şi masca de filtrare W va fi: $W = \{(0,0), (-1,0), (1,0), (0,-1), (-1,-1), (1,-1), (0,1), (-1,1), (1,1)\}$. Pentru acest caz particular, expresia (3.2) devine:

$$v(m, n) = \sum_{k=-1}^{1} \sum_{l=-1}^{1} \frac{f(m - k, n - l)}{9} \tag{3.3}$$

adică, desfăşurat, $v(m,n) = \frac{f(m+1,n-1)}{9} + \frac{f(m+1,n)}{9} + \frac{f(m+1,n+1)}{9} + \frac{f(m,n)}{9} + \frac{f(m,n-1)}{9} + \frac{f(m,n+1)}{9} + \frac{f(m+1,n-1)}{9} + \frac{f(m+1,n)}{9} + \frac{f(m+1,n+1)}{9}$. Acelaşi lucru se poate exprima şi prin specificarea matricială a măştii de filtrare şi marcarea originii acesteia, ca $W = \begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & \boxed{1/9} & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix}$.

Formula de definiţie (3.2) nu este altceva decât suma produselor punct cu punct a coeficienţilor măştii şi a valorilor pixelilor imaginii din zonă de imagine peste care a fost suprapusă masca. Această sumă de produse se calculează pentru fiecare punct al imaginii, deplasând masca. Descrierea algoritmului nu este însă altceva decât descrierea plastică a unei operaţii de convoluţie bidimensională, în care, prin convenţie, masca de filtrare este considerată nucleul de convoluţie. Deplasarea măştii (ferestrei de filtrare) a condus la adoptarea denumirii de tehnică a ferestrei glisante; aplicarea acesteia se poate descrie simplu:

- se plasează originea ferestrei de filtrare (pe rând) în fiecare punct al imaginii şi se selectează punctele imaginii situate în interiorul ferestrei (putem imagina fereastra de filtrare ca fiind o apertură într-o placă opacă; într-o anumită poziţie a acesteia valorile selectate din imagine sunt valorile punctelor ce se "văd" prin apertură).
- pentru fiecare poziţie se face suma produselor punct cu punct coeficient mască - valoare pixel.

Fereastra de filtrare este deci definită de formă (mulţimea W) şi valori (coeficienţii w_{kl}). Cele mai simple ferestre de filtrare sunt cele de formă pătrată, având originea în centru (deci fiind pătrate de dimensiuni impare: 3 x 3, 5 x 5, etc.) - de tipul celei prezentate în exemplul anterior. Fereastra de filtrare are în general o dimensiune mult mai mică decât dimensiunile imaginii (mai sunt numite şi nuclee mici, făcând referinţă la operaţia de convoluţie). Codul care urmează exemplifică această cea mai simplă filtrare, cu un nucleu de dimensiune 3 x 3. Se remarcă alocarea statică a coeficienţilor w ai filtrului (numere reale) şi alocarea dinamică a imaginii rezultat (considerată aici ca având valori întregi). Calculul valorilor pixelilor din imaginea filtrată s-a făcut decupând câte un rând şi o coloană de la extremităţile imaginii originale (pentru a evita efectele de margine, în care masca "debordează" în afara imaginii), acestea fiind completate în imaginea rezultat prin copierea valorilor originale. O altă variantă de evitare a efectelor de margine este bordarea (completarea) imaginii la capete cu câte o linie şi o coloană cu valori nule. De asemenea, se poate constata că în expresia de calcul efectiv indicii tabelului de coeficienţi ai măştii au fost deplasaţi, astfel încât indicele minim să fie 0, şi nu negativ.

```
real w[3][3];
img_out=(int**)malloc(NRLIN*sizeof(int*));
for (i=0;i<NRLIN;i++)
```

```
  img_out[i]=(int*)malloc(NRCOL*sizeof(int));
for (i=0;i<NRCOL;i++)
{ img_out[0][i]=img[0][i];
  img_out[NRLIN-1][i]=img[NRLIN-1][i];}
for (i=0;i<NRLIN;i++)
{ img_out[i][0]=img[i][0];
  img_out[i][NRCOL-1]=img[i][NRCOL-1];}
for (i=1;i<NRLIN-1;i++)
 for (j=1;j<NRCOL-1;j++)
  img_out[i][j]=round(w[1][1]*img[i][j]+w[1][0]*img[i][j-1]+
    w[1][2]*img[i][j+1]+w[0][1]*img[i-1][j]+w[0][0]*img[i-1][j-1]+
    w[0][2]*img[i-1][j+1]+w[2][1]*img[i+1][j]+w[2][0]*img[i+1][j-1]+
    w[2][2]*img[i+1][j+1]);
```

Pentru o imagine pătrată de dimensiune N şi o mască de filtrare pătrată de dimensiune n, numărul de operaţii necesare unei filtrări liniare este de N^2n^2 înmulţiri şi $N^2(n^2-1)$ adunări, adică complexitatea de calcul este $O[N^2]$, atât faţă de dimensiunile imaginii cât şi faţă de dimensiunile ferestrei de filtrare. Aceasta duce la constrângerea practică de a folosi ferestre de filtrare cât mai mici (şi de a încerca descompunerea ferestrelor mari de filtrare într-o succesiune de ferestre mai mici - spre exemplu o filtrare cu un nucleu de 5 x 5, care ar necesita 25 de operaţii pentru fiecare pixel, poate fi echivalată cu două filtrări succesive cu nuclee de 3 x 3 pixeli, care necesită doar 18 operaţii pentru fiecare pixel). În figurile următoare se prezintă efectul unei operaţii de mediere cu un nucleu de 7 x 7: se remarcă faptul că imaginea rezultat este mai neclară şi ceţoasă (efect de *blur*) şi că toate contururile au devenit mai vagi.

Imagine originală

Rezultatul unei filtrări de mediere

3.1 Filtrarea liniară de netezire

Efectul de înceţoşare a unei imagini poate fi considerat şi ca un efect de îmbunătăţire a uniformităţii regiunilor [18]. Aceasta înseamnă că se elimină micile diferenţe dintre valorile pixelilor aparţinând unei aceleiaşi regiuni (zone cu intensitate luminoasă relativ constantă). Acesta este fundamentul metodelor de reducere a zgomotului alb aditiv gaussian (normal) suprapus imaginii: un asemenea zgomot ce afectează o regiune absolut uniformă (în care toţi pixelii ce o formează au aceeaşi valoare) produce variaţia valorilor din interiorul acesteia, deci micşorează uniformitatea. Variaţiile introduse sunt cu atât mai mari cu cât puterea zgomotului este mai mare; pentru un zgomot de medie nulă (aşa cum este zgomotul alb gaussian aditiv) puterea este identică cu varianţa. Teoria proceselor aleatoare [16] arată că, pentru o combinaţie liniară de realizări ale unei variabile aleatoare $y = \sum_{i=1}^{n} a_i x_i$, varianţa noii variabile aleatoare este proporţională cu varianţa variabilei aleatoare din care au provenit realizările particulare: $\sigma_y^2 = \sum_{i=1}^{n} a_i^2 \sigma_x^2$. Deci, prin medierea a n realizări ale unei variabile aleatoare, varianţa este redusă de n ori ($a_i = \frac{1}{n}$).

Măsurile de calitate folosite pentru a caracteriza în mod obiectiv calitatea unei imagini (sau rezultatul unei prelucrări) sunt extensii bidimensionale ale măsurilor de calitate folosite pentru caracterizarea semnalelor unidimensionale. Dacă considerăm f imaginea originală (corectă) şi g imaginea a cărei calitate trebuie determinată (g este considerată că provine din f prin suprapunerea unui zgomot aditiv), rapoartele de calitate cele mai utilizate sunt: raportul semnal zgomot (Signal to Noise Ratio - SNR) (3.4), raportul semnal de vârf zgomot (Peak Signal to Noise Ratio - PSNR) (3.5), eroare pătratică medie (Mean Squared Error - MSE) (3.6) şi eroarea medie absolută (Mean Absolute Error - MAE) (3.7).

$$SNR = 10 \log \frac{\sum_{n=1}^{N} \sum_{m=1}^{M} f^2(n,m)}{\sum_{n=1}^{N} \sum_{m=1}^{M} \left(g(m,n) - f(n,m)\right)^2} \text{ dB} \tag{3.4}$$

$$PSNR = 10 \log \frac{NM \left(\max_{n,m} f(n,m)\right)^2}{\sum_{n=1}^{N} \sum_{m=1}^{M} \left(g(m,n) - f(n,m)\right)^2} \text{ dB} \tag{3.5}$$

$$MSE = \frac{1}{NM} \sum_{n=1}^{N} \sum_{m=1}^{M} \left(g(m,n) - f(n,m)\right)^2 \tag{3.6}$$

$$MAE = \frac{1}{NM} \sum_{n=1}^{N} \sum_{m=1}^{M} |g(m,n) - f(n,m)| \tag{3.7}$$

Fig. 3.1: Imagine degradată de un zgomot aditiv gaussian

Fig. 3.2: Reducerea zgomotului prin filtrarea de mediere

Figurile următoare prezintă o imagine degradată cu un zgomot aditiv, alb, gaussian, de medie nulă (figura 3.1 SNR=16.97 dB, PSNR=22.43 dB, MAE=15.22) şi imaginea filtrată cu un filtru liniar de mediere aritmetică, cu fereastră 3 x 3 (figura 3.2 SNR=19.33 dB, PSNR=24.79 dB, MAE=9.31). Se poate observa atât o calitate vizuală mai bună (regiuni mai uniforme) cât şi factori de calitate mai buni (SNR şi PSNR mai mari, MAE mai mic).

Un caz particular de interes este considerarea regiunilor constante (absolut uniforme) ale imaginii; în aceste regiuni toţi pixelii vor avea aceeaşi valore, şi deci uniformitatea nu mai poate fi îmbunătăţită. În acelaşi timp, operaţia de filtrare de netezire trebuie să păstreze această valoare constantă a pixelilor (pe care o vom nota cu μ); înlocuind în formula de

definiţie a filtrării liniare (3.2), vom obţine că

$$\mu = \sum_{(k,l)\in W} w_{kl}\mu$$

adică:

$$\sum_{(k,l)\in W} w_{kl} = 1 \tag{3.8}$$

Această condiţie (suma coeficienţilor măştii de filtrare să fie unitară) se numeşte condiţia de normare a nucleelor de filtrare de netezire şi defineşte un astfel de nucleu.

Nucleele de filtrare folosite nu trebuie limitate strict la nucleele ce realizează media aritmetică într-o vecinătate pătrată, centrată în punctul considerat. Măştile următoare realizează medieri ponderate: $W_1 = \begin{pmatrix} 1/16 & 1/16 & 1/16 \\ 1/16 & \boxed{1/2} & 1/16 \\ 1/16 & 1/16 & 1/16 \end{pmatrix}$, $W_2 = \begin{pmatrix} 0 & 1/5 & 0 \\ 1/5 & \boxed{1/5} & 1/5 \\ 0 & 1/5 & 0 \end{pmatrix}$, $W_3 = \begin{pmatrix} 0 & 1/8 & 0 \\ 1/8 & \boxed{1/4} & 1/8 \\ 0 & 1/8 & 0 \end{pmatrix}$, $W_4 = \begin{pmatrix} \boxed{1/2} & 1/6 \\ 1/6 & 1/6 \end{pmatrix}$. Nucleul de filtrare poate avea orice formă (deci există o libertate totală în definirea mulţimii W (3.2)); în acelaşi timp însă, practica a demonstrat suficienţa considerării unor forme regulate (pătrate, centrate) pentru mască. Orice mască poate fi extinsă cu puncte ce au ataşat un coeficient nul, pentru a rezulta o mască pătrată centrată; spre exemplu, masca W_4 poate fi scrisă şi ca $W_4 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \boxed{1/2} & 1/6 \\ 0 & 1/6 & 1/6 \end{pmatrix}$; la fel s-a procedat şi cu măştile W_2 şi W_3. Trebuie remarcat că toate aceste măşti respectă condiţia de normare a unui nucleu de netezire (3.8).

3.2 Filtrarea liniară de contrastare

Contrastarea unei imagini are ca obiectiv îmbunătăţirea perceperii vizuale a contururilor obiectelor (îmbunătăţirea detectabilităţii componentelor scenei de-a lungul frontierelor acestora). În principiu, acest deziderat se poate realiza prin modificarea valorilor pixelilor aflaţi de o parte şi de alta a unei frontiere comune.

O experienţă de percepţie vizuală subiectivă ("benzile lui Mach") a pus în evidenţă faptul că sistemul vizual uman are tendinţa de a adânci profilul zonelor de tranziţie dintre regiuni uniforme. Studiul fiziologiei sistemului vizual a demonstrat că acesta se realizează prin prelucrări de tip derivativ ce apar în diferitele etape pe care le parcurge informaţia vizuală; efectul global poate fi descris ca scăderea din semnalul original a unei derivate secunde a acestuia, ponderată corespunzător (aşa cum prezintă figurile 3.3 şi 3.4, pentru cazul unidimensional - secţiunea unei frontiere între regiunile imaginii).

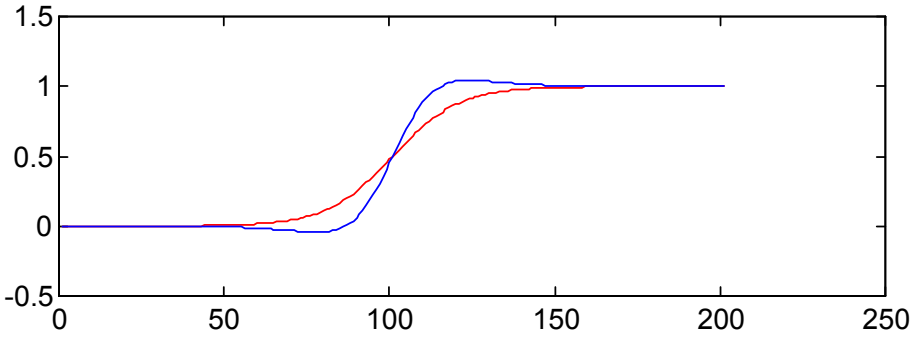

Fig. 3.3: Profilul original de tranziţie şi profilul adâncit prin scăderea derivatei secunde

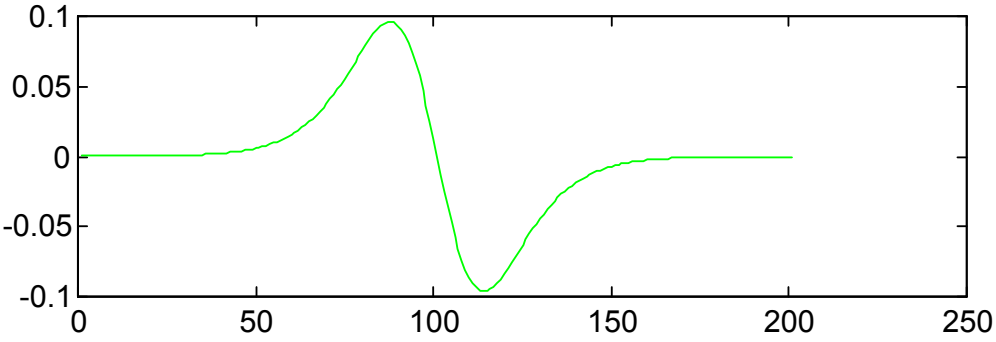

Fig. 3.4: Derivata secundă a profilului original prezentat anterior

Implementarea unei derivate secunde pentru cazul discret va trebui să ia în considerare existenţa a două direcţii de derivare de bază şi modul de definire a derivatei în cazul discret (de exemplu prima derivată pe direcţia orizontală poate fi $f'(m,n) = f(m,n+1) - f(m,n)$ sau $f'(m,n) = f(m,n) - f(m,n-1)$ sau $f'(m,n) = f(m,n+1) - f(m,n-1)$). Măşti obişnuite de implementare a unei derivate secunde bidirecţionale (operator Laplacian) pot fi [9], [19], [5]: $W_5 = \begin{pmatrix} 0 & -1/4 & 0 \\ -1/4 & \boxed{1} & -1/4 \\ 0 & -1/4 & 0 \end{pmatrix}$, $W_6 = \begin{pmatrix} 1/4 & -1/2 & 1/4 \\ -1/2 & \boxed{1} & -1/2 \\ 1/4 & -1/2 & 1/4 \end{pmatrix}$, $W_7 = \begin{pmatrix} -1/8 & -1/8 & -1/8 \\ -1/8 & \boxed{1} & -1/8 \\ -1/8 & -1/8 & -1/8 \end{pmatrix}$.

Pentru un astfel de operator de derivare este esenţial ca răspunsul său pentru pixeli din interiorul unei regiuni absolut uniforme (de valoare μ) să fie nul (derivata unei constante este nulă); pentru aceasta, din formula de definiţie a filtrării liniare (3.2) avem că $0 =$

$\sum_{(k,l)\in W} w_{kl}\mu$, adică:

$$\sum_{(k,l)\in W} w_{kl} = 0 \tag{3.9}$$

Aceasta este condiţia de normare a unui nucleu de filtrare derivativă, pentru care suma coeficienţilor măştii trebuie deci să fie nulă. Se remarcă faptul că nucleele de derivată secundă (operatori laplacieni) prezentate anterior verifică acestă condiţie de normare (3.9). Figurile următoare prezintă o imagine slab contrastată şi varianta sa îmbunătăţită prin contrastare cu nucleul laplacian W_5.

Imagine originală

Contrast îmbunătăţit

3.3 Filtrarea liniară adaptivă

Termenul de adaptiv se referă la capacitatea filtrului de a-şi ajusta comportarea (care este determinată de caracteristicile sale de definiţie) în funcţie de anumite criterii, urmărind optimizarea unor măsuri de calitate. În mod curent, optimizarea măsurilor de calitate se traduce în prelucrarea semnalelor unidimensionale prin minimizarea erorii pătratice medii (sau, corespunzător, maximizarea raportului semnal zgomot). Pe lângă măsurile obiective de calitate, prelucrarea imaginilor adaugă şi o măsură subiectivă: confortul vizual al unui observator, pentru care imaginea dată arată bine, sau mai bine decât o alta. Prin procesul de adaptare, în fiecare nou punct prelucrat structura filtrului este alta, luând în considerare caracteristicile locale pixelului curent prelucrat.

Atâta timp cât (cel puţin teoretic) în fiecare punct al imaginii operaţia este diferită (deoarece se face cu un filtru diferit), filtrarea globală nu mai este liniară (nu mai respectă principiul superpoziţiei (3.1)).

Adaptarea filtrelor liniare nu poate urmări decât două variante: modificarea formei ferestrei de filtrare, sau modificarea coeficienţilor unei ferestre de filtrare de formă fixată.

Exemplul cel mai folosit de modificare a modificare a formei ferestrei de filtrare este filtrul de netezire direcţională adaptivă [9]. Termenul de filtrare direcţională se referă la forma puternic orientată a ferestrei de filtrare (deci fereastra de filtrare nu mai este pătrată, ci va avea o formă liniară, orientată după o anumită direcţie). Pentru fiecare punct al imaginii se pot considera mai multe orientări (deci mai multe direcţii) posibile pentru masca de mediere. Pentru fiecare dintre direcţiile alese se obţine în urma filtrării o valoare. În mod ideal, dorim ca valoarea obţinută prin filtrare să nu difere în mod semnificativ de valoarea iniţială din pixelul considerat. O diferenţă semnificativă poate însemna că punctul curent este situat pe un contur, şi acceptarea acestei valori mult diferite de cea iniţială produce efectul de înceţoşare a imaginii (caracteristic oricărei filtrări de netezire). Soluţia adusă de filtrul adaptiv este de a alege din setul de valori obţinute pentru diferitele ferestre de filtrare pe aceea care este cea mai apropiată de valoarea iniţială din punctul curent. Fragmentul de cod următor prezintă un caz particular de filtrare direcţională adaptivă: ferestrele direcţionale au trei puncte şi sunt alese după cele patru direcţii principale (orizontal, vertical şi cele două diagonale, adică direcţiile orientate la 0°, 45°, 90° şi 135° faţă de orizontală).

```
int val_temp1,val_temp2,out0,out45,out90,out135;
for (i=1;i<NRLIN-1;i++)
 for (j=1;j<NRCOL-1;j++) {
  out0=round((img[i][j]+img[i][j-1]+img[i][j+1])/3);
  out90=round((img[i][j]+img[i-1][j]+img[i+1][j])/3);
  out45=round((img[i][j]+img[i-1][j+1]+img[i+1][j-1])/3);
  out135=round((img[i][j]+img[i-1][j-1]+img[i+1][j+1])/3);
  if (abs(out0-img[i][j])>abs(out90-img[i][j]))
   val_temp1=out90;
  else
   val_temp1=out0;
  if (abs(out45-img[i][j])>abs(out135-img[i][j]))
   val_temp2=out135;
  else
   val_temp2=out45;
  if (abs(val_temp1-img[i][j])>abs(val_temp2-img[i][j]))
   img_out[i][j]=val_temp2;
  else
   img_out[i][j]=val_temp1; }
```

În cod nu a mai fost inclusă alocarea de memorie pentru imaginea de ieşire *img_out* şi nici partea de copiere a valorii marginilor imaginii; se remarcă folosirea a patru variabile suplimentare *out0*, *out45*, *out90*, *out135* care primesc valorile obţinute prin filtrarea direcţională cu masca orientată la 0°, 45°, 90° şi 135°. Variabilele *val_temp1* şi *val_temp2* sunt folosite apoi pentru a calculul valorii celei mai apropiate de valoarea originală.

Ideea de a accepta valoarea produsă de medierea valorilor selectate de fereastra filtrului

numai dacă acestă valoare este suficient de apropiată de valoarea originală se poate aplica şi pentru filtrarea de netezire cu fereastră şi coeficienţi ficşi [5]. În acest caz rezultatul netezirii într-un punct va fi acceptat doar dacă diferenţa faţă de valoarea originală este mai mică decât un prag impus:

$$g(n,m) = \begin{cases} \sum\limits_{(k,l)\in W} w_{kl} f(m-k, n-l), \text{ dacă } \left| f(m,n) - \sum\limits_{(k,l)\in W} w_{kl} f(m-k,n-l) \right| < T \\ f(m,n), \text{ în rest} \end{cases}$$

O variantă clasică de filtrare liniară adaptivă, realizată prin recalcularea coeficienţilor măştii de filtrare în fiecare punct al imaginii este prezentată în [18]. Acest filtru este o combinaţie liniară convexă a imaginii zgomotoase şi a imaginii obţinute din aceasta printr-o filtrare liniară de mediere (cu fereastră constantă), adică: $\hat{f} = \alpha f + \beta \bar{f}$. O combinaţie liniară convexă se obţine doar dacă $\alpha + \beta = 1$. Deci

$$\hat{f} = \alpha f + (1-\alpha)\overline{f} \tag{3.10}$$

Imaginea zgomotoasă f provine dintr-o imagine corectă g, la care s-a adăugat un zgomot alb aditiv de medie nulă n ($\overline{n} = 0$) şi necorelat cu imaginea ($\overline{ng} = \overline{n}\,\overline{g} = 0$): $f = g + n$. Atunci

$$\hat{f} = \alpha(g+n) + (1-\alpha)\overline{(g+n)} = \alpha g + \alpha n + (1-\alpha)\overline{g} \tag{3.11}$$

Ceea ce defineşte filtrul adaptiv este coeficientul α; acesta este calculat local impunând optimizarea unei măsuri de calitate obiective - eroarea pătratică medie între imaginea originală g şi rezultatul filtrării $\widehat{f}$.

$$\varepsilon^2 = \left(\widehat{f} - g\right)^2 = (\alpha n + (\alpha-1)(g-\overline{g}))^2 = \alpha^2 n^2 + 2\alpha(\alpha-1)ng - 2\alpha(\alpha-1)n\overline{g} + (\alpha-1)^2(g-\overline{g})^2$$

$$\overline{\varepsilon^2} = \overline{\alpha^2 n^2 + 2\alpha(\alpha-1)ng - 2\alpha(\alpha-1)n\overline{g} + (\alpha-1)^2(g-\overline{g})^2} = \alpha^2\overline{n^2} + (\alpha-1)^2\overline{(g-\overline{g})^2}$$

$$\overline{\varepsilon^2} = \alpha^2\sigma_n^2 + (\alpha-1)^2\sigma_g^2 \tag{3.12}$$

Formula (3.12) arată că eroarea pătratică medie este suma ponderată dintre varianţa (puterea) zgomotului σ_n^2 şi varianţa imaginii originale σ_g^2. Găsirea coeficientului α optim înseamnă găsirea minimului lui $\overline{\varepsilon^2}$, deci anularea derivatei acestuia.

$$\frac{d\overline{\varepsilon^2}}{d\alpha} = 2\alpha\sigma_n^2 + 2(\alpha-1)\sigma_g^2 = 0$$

$$\alpha = \frac{\sigma_g^2}{\sigma_g^2 + \sigma_n^2} = 1 - \frac{\sigma_n^2}{\sigma_f^2} \tag{3.13}$$

De aici rezultă expresia finală a filtrului:

$$\widehat{f} = \left(1 - \frac{\sigma_n^2}{\sigma_f^2}\right) f + \frac{\sigma_n^2}{\sigma_f^2}\overline{f} \tag{3.14}$$

Se observă că trebuiesc cunoscute puterea de zgomot şi varianţa locală (în fiecare punct) al imaginii zgomotoase. Adaptarea se face între două cazuri limită: dacă puterea de zgomot este mult mai mare decât varianţa valorilor imaginii originale ($\sigma_n^2 \gg \sigma_g^2$) atunci, din prima parte a formulei (3.13) rezultă $\alpha = 0$ şi deci filtrul optim este un simplu filtru de mediere; dacă varianţa valorilor din imagine este mult mai mare decât puterea de zgomot ($\sigma_f^2 \gg \sigma_n^2$, deci este de presupus că punctul curent considerat este situat pe un contur) atunci $\alpha = 1$ şi filtrul optim este un filtru identitate (trece tot). Prezentăm în continuare codul Matlab ce implementează acestă filtrare adaptivă:

```
[NRLIN,NRCOL]=size(img);
for i=2:NRLIN-1
 for j=2:NRCOL-1
   temp=img(i-1:i+1,j-1:j+1);
   temp=temp(:);
   varianta_img=std(temp);
   alfa=1-varianta_zg/varianta_img;
   img_out(i,j)=alfa*img(i,j)+(1-alfa)mean(temp);
 end
end
```

Se folosesc funcţiile Matlab *mean* (pentru a calcula media valorilor selectate de fereastra de filtrare pătrată de 3 x 3) şi *std* (pentru a calcula varianţa locală a imaginii). Se presupune că puterea de zgomot *varianta_zg* este un parametru cunoscut.

În capitolul următor se va introduce o nouă posibilitate de caracterizare a efectelor filtrării liniare a imaginilor: reprezentarea în domeniul frecvenţelor spaţiale (o extensie directă a spectrului semnalelor unidimensionale). De asemenea vom arăta că rămâne valabilă teorema convoluţiei, prin care vom introduce şi o nouă modalitate de implementare a filtrelor liniare: filtrarea în domeniul de frecvenţă.

Capitolul 4

TRANSFORMĂRI INTEGRALE UNITARE DISCRETE

4.1 Generalităţi

Această categorie de prelucrări include operaţii de tip integral – totalitatea pixelilor imaginii iniţiale contribuie la obţinerea valorii fiecărui pixel din imaginea rezultat. În [9] se consideră că termenul de transformări de imagine se referă în mod uzual la o clasă de matrici unitare folosite pentru reprezentarea imaginilor. Orice imagine poate fi reprezentată ca o serie de matrici de bază ortonormale. Pentru o imagine pătrată $\mathbf{U}$[1] de dimensiune N, o dezvoltare în serie este

$$\mathbf{U} = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} \mathbf{A}_{kl}^{*} V(k,l) \tag{4.1}$$

unde $\mathbf{A}_{kl}^{*}$ sunt matricile de bază[2] (de dimensiuni $N \times N$) iar $V(k,l)$ sunt coeficienţii dezvoltării în serie. Exprimând relaţia (4.1) la nivelul fiecărui pixel al imaginii $\mathbf{U}$ obţinem

$$U(m,n) = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} a_{kl}^{*}(m,n) V(k,l) \tag{4.2}$$

Calculul coeficienţilor $V(k,l)$ ai transformării se poate face în condiţiile impuse de orto-

[1] În acest capitol vom folosi notaţiile matriciale şi vectoriale clasice: litere mari, drepte şi groase pentru matrici ($\mathbf{A}$ este o matrice), litere mici, drepte şi groase pentru vectori ($\mathbf{v}$ este un vector) şi litere înclinate pentru elementele vectorilor şi matricilor ($A(i,j)$, $v(m)$).

[2] Numite uneori şi imagini de bază.

gonalitate şi completitudine a matricilor de bază $\mathbf{A}_{kl}^*$ prin:

$$V(k,l) = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} a_{kl}(m,n)U(m,n) \tag{4.3}$$

Mulţimea coeficienţilor transformării $V(k,l)$ formează matricea $\mathbf{V}$, transformata imaginii. Atunci relaţia (4.2) este transformarea directă a imaginii, iar relaţia (4.3), prin care se obţine imaginea din transformata sa, este transformata inversă. Se poate remarca faptul că ambele transformări (directă şi inversă) necesită N^4 operaţii de înmulţire, şi deci o complexitate $O(N^4)$.

Condiţia de ortonormalitate a matricilor de bază se exprimă ca

$$\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} a_{kl}(m,n)a_{k'l'}^*(m,n) = \delta(k-k', l-l'),\ \forall k,l,k',l' \tag{4.4}$$

şi asigură faptul că orice dezvoltare în serie truncheată minimizează eroarea pătratică de aproximare.

Condiţia de completitudine a matricilor de bază se exprimă ca

$$\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} a_{kl}(m,n)a_{kl}^*(m',n') = \delta(m-m', n-n'),\ \forall m,n,m',n' \tag{4.5}$$

şi asigură faptul că baza de matrici folosită este completă.

O transformare de tip (4.2) este deci caracterizată de cei N^4 coeficienţi $a_{kl}(m,n)$, ce pot fi interpretaţi ca valori ai unei funcţii de patru variabile (k,l,m,n), câte două pentru fiecare imagine (iniţială şi transformată). Această funcţie se numeşte nucleul transformării. Prin definiţie, o transfomare se zice separabilă dacă nucleul ei este separabil după perechi de variabile corespondente:

$$a_{kl}(m,n) = a_k(m)b_l(n) = a(k,m)b(l,n) \tag{4.6}$$

Impunând condiţiile (4.4) şi (4.5) acestei noi forme a coeficienţilor transformării, rezultă că matricile $\mathbf{A} = \{a(k,m)\}$ şi $\mathbf{B} = \{b(l,n)\}$ trebuie să fie matrici unitare:

$$\mathbf{A}\mathbf{A}^{*T} = \mathbf{A}^T\mathbf{A}^* = \mathbf{I}_N \tag{4.7}$$

$$\mathbf{B}\mathbf{B}^{*T} = \mathbf{B}^T\mathbf{B}^* = \mathbf{I}_N$$

Pentru o transformare separabilă, relaţiile de definiţie (4.2) şi (4.3) pot fi rescrise sub formă matricială ca:

$$\mathbf{V} = \mathbf{A}\mathbf{U}\mathbf{B}^T \tag{4.8}$$

$$\mathbf{U} = \mathbf{A}^{*T}\mathbf{V}\mathbf{B}^* \tag{4.9}$$

Numărul de înmulţiri necesare pentru a realiza oricare dintre transformările (4.8) sau (4.9) este doar N^3 (deci o complexitate $O(N^3)$). Proprietatea de separabilitate conduce deci la reducerea complexităţii calculelor cu un ordin de mărime; în practică se folosesc numai transformări separabile, pentru care, în plus, $\mathbf{A} = \mathbf{B}$. În acest caz particular, relaţia (4.8) se poate scrie ca

$$\mathbf{V} = \mathbf{AUA}^T = \mathbf{A}(\mathbf{AU}^T)^T = (\mathbf{U}^T\mathbf{A}^T)^T\mathbf{A}^T$$

ceea ce înseamnă că se poate face o transformare unidimensională pe fiecare linie sau coloană a lui $\mathbf{U}$ urmată de aceeaşi transformare pe fiecare coloană sau linie a rezultatului. Astfel, transformările practic utilizate în prelucrarea imaginilor (matricilor) sunt (paradoxal ?) unidimensionale.

Aceste observaţii ne conduc la concluzia că pentru acoperirea cazurilor utilizate în mod curent este suficientă studierea proprietăţilor transformatelor integrale unitare unidimensionale.

4.2 Proprietăţile transformatelor unitare unidimensionale

În cele ce urmează vom considera semnalul de intrare $\mathbf{u} = (u(0), u(1), ..., u(N-1))$ şi transformata sa $\mathbf{v}$, obţinută prin folosirea matricii unitare $\mathbf{A}$. Relaţiile de transformare (directă şi inversă) sunt $\mathbf{v} = \mathbf{Au}$ şi $\mathbf{u} = \mathbf{A}^{*T}\mathbf{v}$.

1. Energia semnalului se conservă printr-o transformare unitară.

$$E_{\mathbf{v}} = \|\mathbf{v}\|^2 = \mathbf{v}^{*T}\mathbf{v} = (\mathbf{Au})^{*T}\mathbf{Au} = \mathbf{u}^{*T}\mathbf{A}^{*T}\mathbf{Au} = \mathbf{u}^{*T}\mathbf{u} = \|\mathbf{u}\|^2 = E_{\mathbf{u}} \qquad (4.10)$$

   Energia vectorului semnal este de fapt lungimea (Euclidiană) a acestuia în spaţiul N-dimensional de reprezentare. Conservarea energiei este deci echivalentă cu conservarea lungimii vectorului, deci transformarea unitară este o rotaţie în spaţiul semnalului. Aceasta înseamnă că baza de reprezentare a lui $\mathbf{u}$ este rotită, iar $\mathbf{v}$ este proiecţia lui $\mathbf{u}$ pe noua bază.

2. Energia medie a semnalului se conservă printr-o transformare unitară.

$$\overline{\mathbf{v}}=\overline{\mathbf{Au}}=\mathbf{A}\overline{\mathbf{u}} \qquad (4.11)$$

$$\overline{E_{\mathbf{v}}} = \overline{\mathbf{v}}^{*T}\overline{\mathbf{v}} = (\mathbf{A}\overline{\mathbf{u}})^{*T}\mathbf{A}\overline{\mathbf{u}} = \overline{\mathbf{u}}^{*T}\mathbf{A}^{*T}\mathbf{A}\overline{\mathbf{u}}=\overline{\mathbf{u}}^{*T}\overline{\mathbf{u}} = \overline{E_{\mathbf{u}}} \qquad (4.12)$$

   Corelaţiei componentelor semnalului se modifică; dacă notăm cu $\mathbf{C}$ matricea de covariaţie atunci:

$$\mathbf{C}_{\mathbf{u}} = \overline{(\mathbf{u}-\overline{\mathbf{u}})(\mathbf{u}-\overline{\mathbf{u}})^{*T}} \qquad (4.13)$$

$$\mathbf{C}_{\mathbf{v}} = \overline{(\mathbf{v}-\overline{\mathbf{v}})(\mathbf{v}-\overline{\mathbf{v}})^{*T}} = \overline{\mathbf{A}(\mathbf{u}-\overline{\mathbf{u}})(\mathbf{u}-\overline{\mathbf{u}})^{*T}\mathbf{A}^{*T}} = \mathbf{A}\mathbf{C}_{\mathbf{u}}\mathbf{A}^{*T} \tag{4.14}$$

Majoritatea transformărilor unitare au tendinţa de a aglomera o mare parte a energiei medii a semnalului în relativ puţini coeficienţi ai transformării. Deoarece energia totală se conservă prin transfomare, mulţi coeficienţi ai transformării vor conţine foarte puţină energie. Energia medie a coeficienţilor transformării va avea o distribuţie neuniformă, chiar dacă în secvenţa de intrare aceasta era uniform distribuită.

Dacă componentele lui $\mathbf{u}$ sunt puternic corelate, coeficienţii transformării vor fi decorelaţi (termenii matricii de covariaţie care nu sunt pe dioagonala principală vor avea valori mici comparativ cu valorile de pe diagonală).

3. Entropia unui vector cu componente aleatoare se conservă printr-o transformare unitară

$$H(\mathbf{u}) = \frac{N}{2}\log_2(2\pi e\,|\mathbf{C}_{\mathbf{u}}|^{1/N}) = \frac{N}{2}\log_2(2\pi e\,|\mathbf{C}_{\mathbf{v}}|^{1/N}) = H(\mathbf{v}) \tag{4.15}$$

Deoarece entropia este o măsură a cantităţii de informaţie (informaţia medie) înseamnă ca transformările unitare păstrează informaţia conţinută în semnal.

4.3 Transformata Fourier discretă

Transformata Fourier este poate cea mai importantă transformare integrală unitară, ce asigură trecerea între spaţiul semnalului şi spaţiul de frecvenţe ale semnalului. După cum semnalul este "clasic" (temporal, şi deci unidimensional) sau cu suport spaţial bidimensional (imagine), spaţiul de frecvenţă marchează frecvenţe propriu-zise sau frecvenţe spaţiale.

Transformata Fourier discretă bidimensională este o transformare separabilă, în care nucleul se poate descompune în termeni identici $\mathbf{A} = \mathbf{B} = \mathbf{F}$. Matricea $\mathbf{F}$ a transformării este o matrice unitară, ce verifică (4.7). Elementele matricii transformării sunt exponenţialele complexe:

$$F(k,n) = \frac{1}{\sqrt{N}}e^{-j\frac{2\pi}{N}kn} = \frac{1}{\sqrt{N}}w_N^{kn} \tag{4.16}$$

Separabilitatea ne permite deci să studiem majoritatea proprietăţilor transformării pe cazul unidimensional, urmând ca rezultatele să fie uşor extinse pentru cazul bidimensional. Folosind notaţiile introduse în secţiunea 4.2, putem scrie deci transformata Fourier unidimensională directă ca:

$$v(k) = \sum_{n=0}^{N-1} u(n)w_N^{kn},\ k = \overline{0, N-1} \tag{4.17}$$

iar transformarea inversă ca

$$u(n) = \frac{1}{N} \sum_{k=0}^{N-1} v(k) w_N^{-kn},\ n = \overline{0, N-1} \tag{4.18}$$

Relaţiile se extind imediat la cazul bidimensional:

$$v(k,l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} u(m,n) w_N^{kn+ml},\ k,l = \overline{0, N-1} \tag{4.19}$$

$$u(m,n) = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} v(k,l) w_N^{-(kn+ml)},\ m,n = \overline{0, N-1} \tag{4.20}$$

Trebuie totuşi remarcat că formele prezentate ale transformărilor nu mai sunt unitare (apare o asimetrie între transformarea directă şi cea inversă prin factorul de scalare de tip $1/N$). Matematic, definiţia (4.16) este corectă, dar în practică se folosesc formele neunitare prezentate în (4.17), (4.18) şi (4.19), (4.20).

4.3.1 Proprietăţile fundamentale ale transformatei Fourier

Dintre numeroasele proprietăţi ale transformatei Fourier [9], [19] ne vom referi doar la două proprietăţi fundamentale: simetria centrală a spectrului de frecvenţă (având drept consecinţă importantă faptul că este necesar acelaşi spaţiu de memorie pentru a reprezenta fie spectrul unei imagini, fie imaginea propriu-zisă) şi teorema convoluţiei circulare (care face legătura între filtrarea în domeniul spaţial şi în domeniul de frecvenţă).

Transformata Fourier a unei secvenţe (matrici) reale este complex conjugată faţă de mijlocul său [9]. Aceasta înseamnă că

$$v(\frac{N}{2} - k) = v^*(\frac{N}{2} + k),\ k = \overline{0, \frac{N}{2}} \tag{4.21}$$

$$v(\frac{N}{2} - k, \frac{N}{2} - l) = v^*(\frac{N}{2} + k, \frac{N}{2} + l),\ k,l = \overline{0, \frac{N}{2}}$$

Relaţiile arată că există o simetrie centrală a spectrelor de frecvenţă, în centru aflându-se o valoare reală. Dar singura valoare reală din spectre este cea ce corespunde componentei de fecvenţă nulă (componenta medie), şi deci este necesară o interschimbare a jumătăţilor de spectru (în cazul secvenţelor unidimensionale) sau a sferturilor de spectru (în cazul imaginilor), astfel încât componenta de frecvenţă nulă să fie în centru. În figura 4.1 este reprezentat modulul spectrului de frecvenţă al imaginii "lena", reprezentat cu 256 nivele de gri (valorile au fost scalate după o reprezentare logaritmică) astfel încât valorile mai mari corspund unei nuanţe mai deschise. Se observă în partea dreaptă spectrul central simetrizat, în care valorile maxime (corespunzâd frecvenţelor cele mai mici, inclusiv

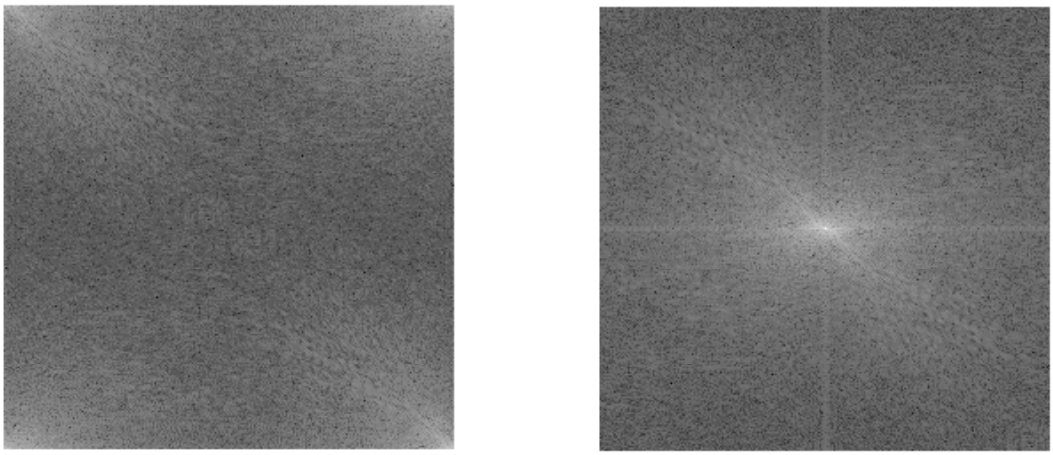

Fig. 4.1: Spectrele de frecvenţă (Fourier) ale imaginii "lena", înainte şi după aranjarea cu simetrie centrală.

zero) se găsesc în centrul imaginii, iar în partea stângă spectrul original obţinut în urma transformării Fourier, în care maximele se găsesc pe cele patru colţuri.

Ca şi în cazul unidimensional, şi pentru imagini legătura dintre frecvenţele spaţiale şi obiectele din domeniul spaţial (imagine) se păstrează: frecvenţele spaţiale ridicate corespund detaliilor, obiectelor mici, contururilor, zgomotului. În figura 4.2 este prezentat spectrul de frecvenţă al imaginii "lena" degradată de zgomot impulsiv de tip sare şi piper (figura 5.1); zgomotul impulsiv corespunde apariţiei a numeroase detalii (obiecte mici – punctele de zgomot) şi variaţii ale nivelului de gri. Efectul evident este acela de mărire a valorilor din spectru corespunzătoare frecvenţelor înalte şi, corespunzător, diminuarea importanţei relative a frecvenţelor joase.

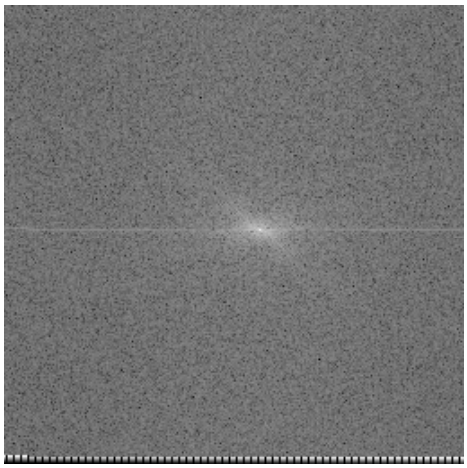

Fig. 4.2: Spectrul de frecvenţă al imaginii "lena" degradată de zgomot impulsiv.

Prelucrarea imaginilor (fie ele scalare sau vectoriale) presupune executarea, prin intermediul unui bloc funcţional ce poate fi presupus iniţial de tip "cutie neagră", a unei operaţii de tip *Image In, Image Out* [2] (spre deosebire de tehnicile de analiză ce pot fi caracterizate ca operaţii *Image In, Description Out*). Transformarea imaginii de intrare în imaginea rezultat de ieşire (ce prezintă aceleaşi caracteristici majore, simbolice şi perceptuale[3]) se numeşte în mod generic filtrare[4]. Funcţia unui filtru este de a transforma un semnal dat într-un alt semnal, mai potrivit unei anume aplicaţii date.

Definiţiile filtrării (sau ale dispozitivului care o realizează, filtrul) ce se regăsesc în dicţionarele lingvistice de uz general fac apel la noţiunea de semnal electric şi de frecvenţe asociate componentelor acestuia. Descrierea acţiunii filtrului este descrierea unei operaţii liniare efectuate în domeniul frecvenţelor. Extinderea acestor definiţii la cazul imaginilor presupune în primul rând definirea spectrului de frecvenţă asociat unui semnal cu suport [spaţial] multidimensional şi, implicit, introducerea noţiunii de frecvenţă spaţială. Utilizarea unei prelucrări în domeniul frecvenţelor presupune apoi identificarea benzilor de frecvenţă corepunzătoare entităţilor ce se doresc păstrate, respectiv eliminate, asociere ce nu este întotdeauna simplă. În practică, aceasta înseamnă determinarea unui spectru de frecvenţe transformat, care apoi este transformat prin transformata Fourier inversă în imaginea filtrată – aceasta este filtrarea în domeniul frecvenţă.

Filtrarea liniară a imaginilor se bazează pe convoluţia [circulară] între imaginea de prelucrat şi un nucleu de filtrare (a se vedea capitolul 3); teoria transformatelor unitare [9] arată că o asemenea operaţie este echivalentă unui produs între spectrul Fourier al imaginii şi spectrul Fourier al nucleului de filtrare; aceasta este teorma convoluţiei. În cazul unidimensional, dacă secvenţele $\mathbf{x}$ şi $\mathbf{y}$ au aceeaşi lungime, şi $\circledast$ este operatorul de convoluţie circulară (definit în (4.23)) atunci:

$$\text{Fourier}(\mathbf{x} \circledast \mathbf{y}) = \text{Fourier}(\mathbf{x})\,\text{Fourier}(\mathbf{y}) \tag{4.22}$$

$$\mathbf{x} \circledast \mathbf{y}(n) = \sum_{k=0}^{N-1} x\left((n-k) \bmod N\right) y(k),\ n = \overline{0, N-1} \tag{4.23}$$

Teorema convoluţiei oferă deci instrumentul prin care se pot echivala operaţiile de filtrare realizate în domeniile spaţial (filtrarea liniară, prezentată în capitolul 3) şi de frecvenţă. Aceasta înseamnă că un nucleu mic de convoluţie (mască de filtrare) este bordat cu zerouri până la obţinerea dimensiunii imaginii (teorema convoluţiei circulare se referă la

[3]Prin prisma acestei definiţii, transfomările unitare ale imaginilor (deci reprezentarea într-un spaţiu spectral) nu sunt filtrări; spectrul unei imagini, deşi este echivalent acesteia din punct de vedere informaţional, nu are aceleaşi caracteristici perceptuale.

[4]Filtrul este un sistem de circuite electrice, sonore, etc. cu care se selectează dintr-un complex de oscilaţii cu frecvenţe diferite, oscilaţiile cu frecvenţele cuprinse între anumite limite. (Dicţionarul Explicativ al Limbii Române, 1995)

Filtrul este un dispozitiv ce amortizează selectiv oscilaţiile cu anumite frecvenţe şi nu afectează oscilaţiile având alte frecvenţe. (Webster Encyclopedic Unabridged Dictionary, 1995)

Filtrul este un dispozitiv destinat favorizării sau inhibării trecerii anumitor componente de frecvenţă a unui semnal electric. (Larousse, 1994)

secvenţe de aceeaşi lungime) şi apoi transformat Fourier; ceea ce se obţine este răspunsul în frecvenţă al filtrului. Figura 4.3 prezintă răspunsul în frecvenţă a filtrului de mediere aritmetică realizat cu un nucleu pătrat, centrat de 3 x 3; se remarcă comportamentul de tip filtru trece jos. Acest comportament este leagt de efectul de "blur" al filtrului de mediere, pus în evidenţă prin reducerea vizibilităţii contururilor imaginii, şi deci înlăturarea frecvenţelor înalte ce le sunt asociate.

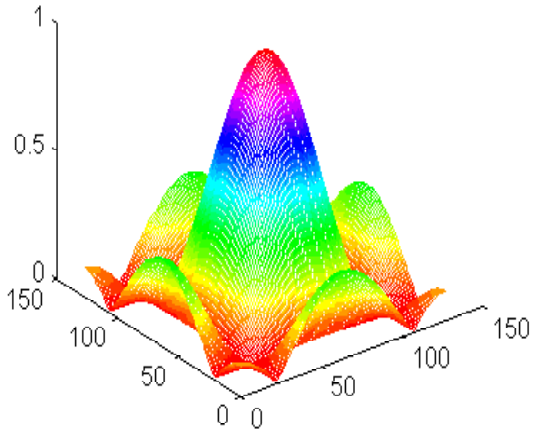

Fig. 4.3: Răspunsul în frecvenţă a unui nucleu 3 x 3 de mediere aritmetică.

4.3.2 Transformata Fourier rapidă

Implementarea directă a relaţiei de definiţie a transformării Fourier discrete unidimensionale (4.17) necesită evident N^2 înmulţiri complexe şi $N(N-1)$ adunări, ceea ce duce la o complexitate $O(N^2)$. Realizarea unei implementări mai rapide a transformării Fourier (FFT - *Fast Fourier Transform*) a constituit unul dintre momentele cele mai importante pentru domeniul prelucrării digitale a semnalelor. Tehnica iniţială a fost divizarea în timp: cele N eşantioane ale secvenţei se împart în două, după indicii pari şi respectiv impari:

$$v(k) = \sum_{n=0}^{N-1} u(n) \exp(-j\frac{2\pi}{N}kn) =$$

$$= \sum_{n=0}^{N/2-1} u(2n) \exp(-j\frac{2\pi}{N}k \cdot 2n) + \sum_{n=0}^{N/2-1} u(2n+1) \exp(-j\frac{2\pi}{N}k(2n+1)) =$$

$$= \sum_{n=0}^{N/2-1} u(2n) \exp(-j\frac{2\pi k}{N/2}2n) \exp(j\frac{2\pi k}{N}2n) + \exp(-j\frac{2\pi k}{N}) \sum_{n=0}^{N/2-1} u(2n+1) \exp(-j\frac{2\pi k}{N/2}2n) =$$

$$= \sum_{n=0}^{N/2-1} u_{par}(n) \exp(-j\frac{2\pi k}{N/2}n) + \exp(-j\frac{2\pi k}{N}) \sum_{n=0}^{N/2-1} u_{impar}(n) \exp(-j\frac{2\pi k}{N/2}n) =$$

$$= v_{par}(k) + \exp(-j\frac{2\pi k}{N}) v_{impar}(k) \tag{4.24}$$

În (4.24) am notat cu $\mathbf{u}_{par}$ secvenţa obţinută din toate valorile pare ale secvenţei iniţiale, iar cu $\mathbf{u}_{impar}$ secvenţa obţinută din toate valorile impare ale secvenţei iniţiale. Această relaţie exprimă posibilitatea construirii transformării Fourier a secvenţei $\mathbf{u}$ de lungime N prin transformările Fourier ale unor jumătăţi de secvenţă (de lungime $N/2$) – $\mathbf{v}_{par}$ şi $\mathbf{v}_{impar}$.

Dacă notăm cu $T(N)$ timpul de calcul al transformatei Fourier a secvenţei de lungime N, atunci avem:

$$T(N) = 2T\left(\frac{N}{2}\right) + N \tag{4.25}$$

În mod evident $T(1) = 1$ (transformata Fourier a secvenţei de lungime 1 este identică cu secvenţa); dezvoltând ecuaţia recurentă din (4.25) obţinem:

$$T(N) = 2T\left(\frac{N}{2}\right) + N = 2\left(2T\left(\frac{N}{4}\right) + \frac{N}{2}\right) + N = 4T\left(\frac{N}{4}\right) + 2N =$$

$$= ... = 2^i T\left(\frac{N}{2^i}\right) + iN \tag{4.26}$$

La limită, când i este ales astfel ca $N = 2^i$ vom avea:

$$T(N) = NT(1) + N\log_2 N = N\log_2 N + N$$

ceea ce este echivalent cu o complexitate de $O(N \log N)$. Sporul de viteză obţinut prin FFT faţă de implementarea clasică este deci proporţional cu $N/\log_2 N$ (deci aproape un ordin de mărime pentru o lungime tipică de secvenţă $N = 256$).

Relaţia de implementare de bază (4.24) mai poate fi scrisă şi ca:

$$v(k) = v_{par}(k) + w_N^k v_{impar}(k),\ k = \overline{0, ..., \frac{N}{2} - 1} \tag{4.27}$$

$$v(k) = v_{par}(k) - w_N^k v_{impar}(k),\ k = \overline{\frac{N}{2}, ..., N - 1}$$

ceea ce semnifică faptul că două valori ale transformărilor Fourier ale secvenţelor pe jumătate sunt combinate liniar pentru a genera două valori, simetrice faţă de centru, ale secvenţei originale. Blocul ce realizează aceste prelucrări este numit celulă "*butterfly*", caracterizat de o operaţie de înmulţire-adunare[5] (figura 4.4).

[5] Operaţia de îmulţire-adunarte este una dintre operaţiile de bază (cablate) din setul de instrucţiuni al procesoarelor de semnal (DSP).

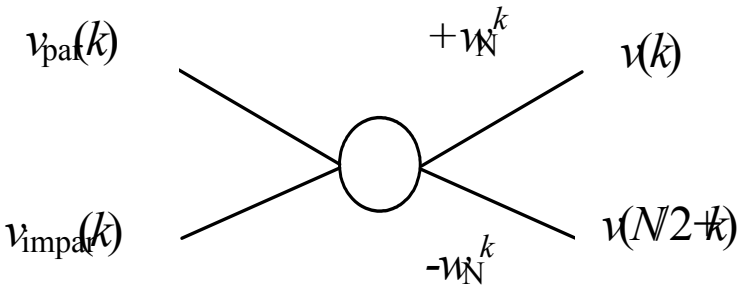

Fig. 4.4: Celula de bază butterfly

Construind etapele succesive de înjumătăţire a lungimii secvenţei, pentru o secvenţă de lungime $N = 8$ se ajunge la schema din figura 4.5. Se remarcă reordonarea eşantioanelor secvenţei de intrare după ordinea de bit inversă *bit-reverse* (forma binară a noului indice este forma binară a vechiului indice reflectată).

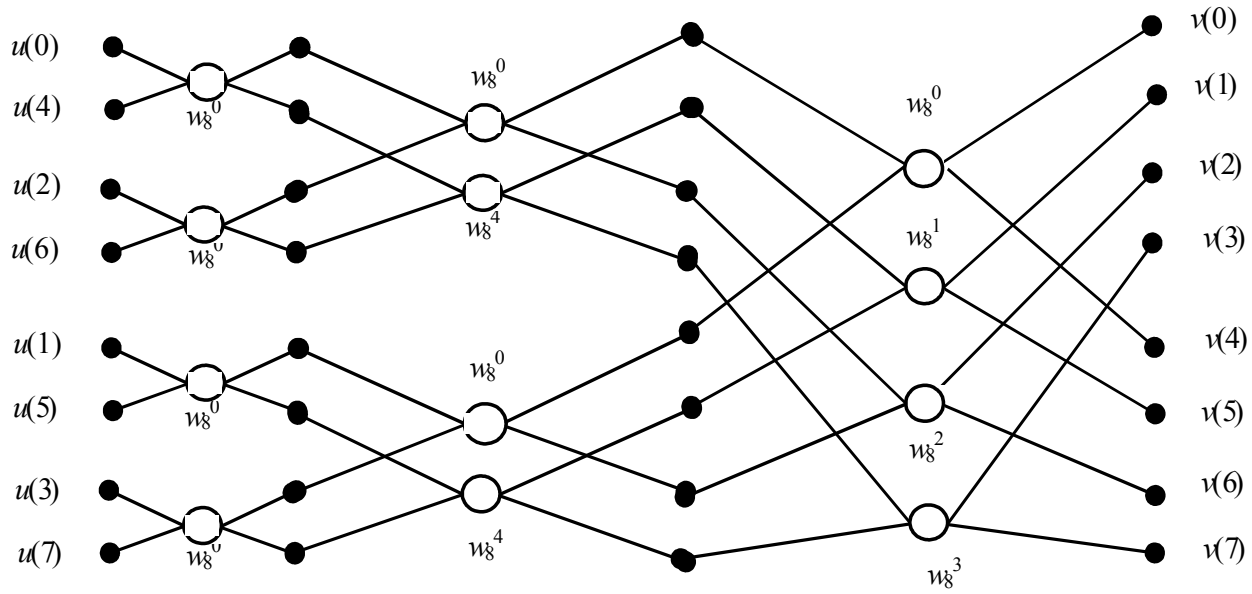

Fig. 4.5: Schema FFT pentru o secvenţă de lungime 8.

Implementările clasice ale FFT folosesc o reordonare *in-place* a secvenţei de intrare (în acelaşi spaţiu de memorie, prin interschimbări) pentru numere complexe (scrise în ordinea parte reală, parte imaginară). Acelaşi cod este folosit atât pentru transformata directă cât şi pentru transformata inversă, doar prin modificarea unui parametru de semn ce intervine la calculul coeficienţilor Fourier w_N. Rezultatul (transformata) ia locul datelor

de intrare, în acelaşi spaţiu de memorie (deci din nou o implementare *in-place*). În continuare prezentăm codul C din [13].

```
void fft(float data,integer nn,isign)
integer m,n,mmax,i,j,step;

n=nn<<1;
j=1;
for (i=1;i<n;i+=2){
  if (j>i) {
    swap(data[j],[data[i]);
    swap(data[j+1],[data[i+1]);}}
m=n>>1;
while (m>=2 && j>m) {
  j-=m;
  m>>=1;}
j+=m;}

mmax=2;
while (n>mmax) {
  step=2*mmax;
  teta=2*PI/(isign*mmax);
  wtemp=sin(teta/2);
  wpr=-2*wtemp*wtemp;
  wpi=sin(teta);
  wr=1;
  wi=0;
  for (m=1;m<mm1x;m+=2) {
    for (i=m;i<=n;i+=step) {
      j=i+mmax;
      tempr=wr*data[j]-wi*data[j+1];
      tempi=wr*data[j+1]+wi*data[j];
      data[j]=data[i]-tempr;
      data[j+1]=data[i+1]-tempi;
      data[i]+=tempr;
      data[i+1]+=tempi;}
    wr=(wtemp=wr)*wpr-wi*wpi+wr;
    wi=wi*wpr+wtemp*wpi+wi;}
  mmax=step;}
```

Prima parte a codului realizează reordonarea secevenţei în ordinea inversă de bit. În partea a doua a codului se realizează calculul efectiv al celulelor *butterfly*, corespunzând transformărilor unor secvenţe de dimensiune *mmax*, şi deci preluării unor date situate la

distanţa *mmax* între indici (vizibil şi în schema din figura 4.5).

Transformata rapidă bidimensională implică realizarea câte unei transformări unidimensionale pentru fiecare linie şi coloană a imaginii (conform principiului separabilităţii); aceasta conduce la o complexitate $O(N^2 \log N)$.

4.4 Alte transformări

În cele ce urmează vom prezenta pe scurt alte transformări unitare folosite în prelucrarea imaginilor; apariţia acestora s-a datorat adaptării mai bune decât transformata Fourier la compactarea energiei şi decorelarea spectrală a anumitor clase de semnale.

4.4.1 Transformata cosinus

Transformata cosinus este o transformată unitară separabilă, caracterizată de $\mathbf{A} = \mathbf{B} = \mathbf{C}$. Elementele matricii $\mathbf{C}$ sunt date de

$$C(k,n) = \frac{1}{\sqrt{N}} \begin{cases} 1, \text{ dacă } k = 0 \\ \sqrt{2}\cos\frac{(2n+1)\pi k}{2N}, \text{ dacă } k = \overline{1, N-1} \end{cases}, n = \overline{0, N-1} \tag{4.28}$$

Transformările directă şi inversă a unei secvenţe $\mathbf{u}$ sunt definite ca:

$$v(k) = \alpha(k) \sum_{n=0}^{N-1} u(n) \cos\frac{(2n+1)\pi k}{2N},\ k = \overline{0, N-1} \tag{4.29}$$

$$u(n) = \sum_{k=0}^{N-1} \alpha(k) v(k) \cos\frac{(2n+1)\pi k}{2N},\ n = \overline{0, N-1} \tag{4.30}$$

unde

$$\alpha(k) = \frac{1}{\sqrt{N}} \begin{cases} 1,\ k = 0 \\ \sqrt{2},\ k \neq 0 \end{cases} \tag{4.31}$$

Una dintre proprietăţile importante ale transformatei cosinus se referă la legătura acesteia cu transformata Fourier: transformata cosinus a unei secvenţe nu este partea reală a transformatei Fourier a aceleiaşi secvenţe. Transformata cosinus se poate obţine prin transformarea Fourier a unei secvenţe rearanjate sau a unei secvenţe dublate şi simetrizate central. Fie N par; şi definim

$$\begin{matrix} \widetilde{u_1}(n) = u(2n) \\ \widetilde{u_1}(N-n-1) = u(2n+1) \end{matrix}, n = \overline{0, \frac{N}{2} - 1} \tag{4.32}$$

$$\widetilde{u_2}(n) = \begin{cases} u(N-n-1),\ 0 \leqslant n < N \\ u(n-N),\ N \leqslant n < 2N \end{cases} \tag{4.33}$$

Aceasta înseamnă că secvenţa $\widetilde{\mathbf{u}_1}$ este $u(0), u(2), u(4), ..., u(N-2), u(N-1), u(n-3), ..., u(3), u(1)$, iar secvenţa $\widetilde{\mathbf{u}_2}$ este $u(N-1), u(N-2), ..., u(1), u(0), u(0), u(1), ..., u(N-2), u(n-1)$. Atunci:

$$COS(\mathbf{u}) = \text{Re}\left[\alpha(k)\exp(-\frac{j\pi k}{2N})\,\text{Fourier}(\widetilde{\mathbf{u}_1})\right] \tag{4.34}$$

$$COS(\mathbf{u}) = \text{Fourier}(\widetilde{\mathbf{u}_2})\exp(-\frac{\pi k}{2N}),\ k = \overline{0, ..., N-1} \tag{4.35}$$

O consecinţă imediată a acestor relaţii este posibilitatea realizării unei transformări cosinus rapide (de complexitate $O(N \log N)$) folosind transformata Fourier rapidă.

O altă proprietate importantă a transformatei cosinus este accea că vectorii bazei cosinus (coloanele matricii $\mathbf{C}$) sunt vectori proprii pentru orice matrice simetrică tridiagonală de forma:

$$\mathbf{Q} = \begin{pmatrix} 1-\rho & -\rho & 0 & ... & ... & 0 \\ -\rho & 1-\rho & -\rho & 0 & ... & 0 \\ 0 & -\rho & 1-\rho & -\rho & ... & 0 \\ ... & ... & ... & ... & ... & ... \\ 0 & ... & 0 & -\rho & 1-\rho & -\rho \\ 0 & ... & ... & 0 & -\rho & 1-\rho \end{pmatrix} \tag{4.36}$$

Dacă notăm cu $\mathbf{c}_k$ coloana k a matricii transformării cosinus, atunci:

$$\mathbf{Q}\mathbf{c}_k = \lambda_k \mathbf{c}_k$$

Această proprietate revelă faptul că transfomata cosinus se comportă precum transformata Karhunen-Loeve pentru orice clasă de semnale a căror matrice de covariaţie este de forma (4.36). Dar această formă este tipică secvenţelor staţionar Markov de ordinul 1, cu parametru de corelaţie mare ($\rho \to 1$), model adeseori folosit în cazul imaginilor. Deci transformata cosinus este similară transformării optime de decorelare Karhunen-Loeve pentru o clasă largă de imagini naturale (alte comentarii referitoare la această proprietate şi la aplicaţiile ei practice se găsesc în secţiunea 7.2.2, referitoare la compresia imaginilor cu transformate).

4.4.2 Transformata sinus

Transformata sinus este o transformată unitară separabilă, caracterizată de $\mathbf{A} = \mathbf{B} = \mathbf{S}$. Elementele matricii $\mathbf{S}$ sunt date de

$$S(k,n) = \sqrt{\frac{2}{N+1}}\sin\frac{\pi(k+1)(n+1)}{N+1},\ k,n = \overline{0, N-1} \tag{4.37}$$

Transformările directă şi inversă a unei secevenţe $\mathbf{u}$ sunt definite ca:

$$v(k) = \sqrt{\frac{2}{N+1}} \sum_{n=0}^{N-1} u(n) \sin \frac{\pi(k+1)(n+1)}{N+1},\ k = \overline{0, N-1} \tag{4.38}$$

$$u(n) = \sqrt{\frac{2}{N+1}} \sum_{k=0}^{N-1} v(k) \sin \frac{\pi(k+1)(n+1)}{N+1},\ n = \overline{0, N-1} \tag{4.39}$$

Una dintre proprietăţile importante ale transformatei sinus se referă la legătura acesteia cu transformata Fourier: transformata sinus a unei secvenţe nu este partea imaginară a transformatei Fourier a aceleiaşi secvenţe. Transformata sinus se poate obţine prin transformarea Fourier a extensiei antisimetrice a secvenţei. Fie N par; şi definim

$$\begin{array}{l} \widetilde{u}(0) = \widetilde{u}(N+1) = 0 \\ \widetilde{u}(n) = -u(N-n) \quad , n = \overline{1, N} \\ \widetilde{u}(N+2+n) = u(n) \end{array} \tag{4.40}$$

Aceasta înseamnă că secvenţa $\widetilde{\mathbf{u}}$ este $0, -u(N-1), -u(N-2), ..., -u(1), -u(0), 0, u(0), u(1), ..., u(N-1)$. Atunci

$$SIN(\mathbf{u}) = \text{Im}\left[-j(-1)^k \,\text{Fourier}(\widetilde{\mathbf{u}})\right] \tag{4.41}$$

O consecinţă imediată a acestei relaţii este posibilitatea realizării unei transformări sinus rapide (de complexitate $O(N \log N)$) folosind transformata Fourier rapidă.

O altă proprietate importantă a transformatei sinus este aceea că vectorii bazei sinus (coloanele matricii $\mathbf{S}$) sunt vectori proprii pentru orice matrice simetrică tridiagonală de forma:

$$\mathbf{Q} = \begin{pmatrix} 1 & -\rho & 0 & ... & 0 \\ -\rho & 1 & 0 & ... & 0 \\ ... & ... & ... & ... & ... \\ 0 & ... & 0 & 1 & -\rho \\ 0 & ... & 0 & -\rho & 1 \end{pmatrix} \tag{4.42}$$

Dacă notăm cu $\mathbf{s}_k$ coloana k a matricii transformării sinus, atunci:

$$\mathbf{Q}\mathbf{s}_k = \lambda_k \mathbf{s}_k$$

Această proprietate revelă faptul că transfomata sinus se comportă precum transformata Karhunen-Loeve pentru orice clasă de semnale a căror matrice de covariaţie este de forma (4.42). Dar această formă este tipică secvenţelor staţionar Markov de ordinul 1, cu parametru de corelaţie $\rho \in [-0.5; 0.5]$.

Capitolul 5

FILTRAREA NELINIARĂ A IMAGINILOR

Operaţiile de filtrare studiate până în prezent au fost caracterizate esenţial (din punct de vedere matematic) de către principiul liniarităţii (sau superpoziţiei). Istoric, acestea au fost primele operaţii utilizate, având un suport teoretic solid şi desigur, având mai multe caracteristici utile. O mare parte a aplicaţiilor curente de prelucrare a imaginilor probabil că se pot rezolva suficient de precis folosind doar operaţii liniare. Limitările operaţiilor liniare de filtrare apar însă în momentul în care se consideră zgomote ce nu au distribuţie normală, şi, mai mult, nu mai sunt aditive.

Exemplul cel mai simplu de astfel de zgomot este zgomotul impulsiv, caracterizat de impulsuri pozitive şi negative de amplitudine maxim posibilă (relativ la numărul de nivele de cuantizare ale imaginii) care afectează prin înlocuire o parte din pixelii imaginii. Aceasta înseamnă că imaginea va fi uniform acoperită de puncte foarte închise (negre) şi foarte deschise (albe), rezultatul fiind ceea ce se numeşte zgomot de tip sare şi piper (vezi figura 5.1). Asemenea erori apar în porţiunile de achiziţie şi transmisiune ale sistemului de prelucrarea şi analiza imaginilor (din cauza erorilor în convertorul analog-digital, din cauza erorilor de înregistrare pe mediile magnetice de stocare sau de pe canalul de transmisiune); este posibil ca asemenea valori să apară şi ca urmare a unui zgomot aditiv caracterizat de o distribuţie "cu coadă lungă" - deci care asigură probabilităţi nenule de apariţie a unor valori mult diferite de medie. Aceste valori aberante (mult mai mari sau mai mici decât valorile normale ale regiunii în care apar) sunt numite "outlier": valori extreme sau erori grosiere.

Dacă o asemena imagine ar fi filtrată liniar (cu un filtru de mediere sau poate cu un filtru de reliefare) rezultatul ar fi o accentuare a punctelor de zgomot şi o corupere a punctelor cu valori corecte din jurul lor.

Zgomotul impulsiv nu este singurul model de degradare neliniară: există zgomote de-

Fig. 5.1: Imagine afectată de zgomot impulsiv; 10% dintre pixeli au valorile modificate

pendente de valoarea imaginii din pixelul afectat, precum şi compuneri neaditive ale zgomotului cu imaginea originală (multiplicativă, convolutivă) [9].

Este deci evident că se impune considerarea şi a altor metode de filtrare, care nu mai respectă principiul superpoziţiei, şi deci sunt neliniare. O clasă esenţială de astfel de metode de filtrare sunt cele bazate pe ordonare - rank order filtering [12]. Ideea esenţială este aceea că operaţia de ordonare plasează valorile aberante (extremale) la capetele şirului de valori (deci într-o zonă bine definită), permiţând astfel identificarea şi eliminarea acestora.

5.1 Filtrarea de ordine

Filtrele de ordine sunt operatori locali: filtrul este definit de o fereastră (mască), care selectează din imaginea de prelucrat un număr de vecini ai pixelului curent (se respectă deci acelaşi model de prelucrare al ferestrei glisante, întâlnit şi la filtrările liniare în domeniul spaţial). Valorile selectate de fereastra de filtrare sunt apoi ordonate crescător; dacă valorile selectate de o fereastră cu n poziţii sunt $\{x_1, x_2, ..., x_n\}$, atunci acelaşi set de valori, ordonate crescător este $\{x_{(1)}, x_{(2)}, ..., x_{(n)}\}$. Valorile $x_{(i)}$ se numesc statistici de ordine de ordinul i şi au proprietatea

$$x_{(1)} \leqslant x_{(2)} \leqslant ... \leqslant x_{(n)} \tag{5.1}$$

În mod evident statistica de ordinul 1, $x_{(1)}$, este valoarea minimă, iar statistica de ordinul n, $x_{(n)}$, este valoarea maximă.

Ieşirea filtrului de ordine de ordin k este statistica de ordinul k, cu $k \in [1; n]$:

$$\text{rank}_k\{x_1, x_2, ..., x_n\} = x_{(k)} \tag{5.2}$$

În acest fel, ieşirea unui filtru de ordine este una dintre valorile selectate de fereastra de filtrare, şi deci nu se crează la ieşire valori noi (aşa cum se întâmplă la filtrarea liniară); acesta este un avantaj atât în ceea ce priveşte păstrarea valorilor originale din imagine, cât şi din punctul de vedere al simplificării calculelor (nu mai sunt necesare operaţii cu numere reale, ce trebuie apoi rotunjite sau truncheate la numere naturale).

Filtrele de ordine nu sunt liniare din cauza operaţiei de ordonare (ce nu respectă principiul superpoziţiei); în general putem scrie:

$$\text{rank}_k\{x_1+y_1, x_2+y_2, ..., x_n+y_n\} \neq \text{rank}_k\{x_1, x_2, ..., x_n\} + \text{rank}_k\{y_1, y_2, ..., y_n\} \quad (5.3)$$

Filtrele de ordine comută însă cu operaţiile de modificare liniară a valorilor:

$$\text{rank}_k\{\alpha x_1+\beta, \alpha x_2+\beta, ..., \alpha x_n+\beta\} = \alpha\,\text{rank}_k\{x_1, x_2, ..., x_n\} + \beta \quad (5.4)$$

Din punct de vedere statistic, interesează funcţiile de distribuţie ale ieşirii filtrului (funcţie de densitate de probabilitate, funcţie de repartiţie) atunci când funcţiile de distribuţie ale valorilor de intrare sunt presupuse cunoscute. Modelul folosit în general se bazează pe ipoteza de independenţă şi distribuţie identică a valorilor selectate de fereastra filtrului (ceea ce conduce, implicit, la ipoteza că valorile pixelilor imaginii sunt variabile aleatoare independente, identic distribuite[1]). Dacă notăm cu $f(x)$ şi $F(x)$ funcţiile de densitate de probabilitate şi respectiv repartiţie a valorilor pixelilor din imaginea ce se filtrează cu un filtru de ordine de ordin k cu fereastră de n elemente, atunci ceea ce interesează este funcţia de densitate de probabilitate a ieşirii filtrului de ordine, $f_k(x)$.

Fie t o valoare oarecare din domeniul de variaţie al variabilelor aleatoare; atunci probabilitatea evenimentului ca ieşirea filtrului de ordine să se găsească în intervalul $[t; t+dt]$ este $f_k(t)dt$. Evenimentul de interes este deci

$$t \leqslant \text{rank}_k\{x_1, x_2, ..., x_n\} \leqslant t+dt \quad (5.5)$$

adică

$$t \leqslant x_{(k)} \leqslant t+dt \quad (5.6)$$

Statisticile de ordine sunt obţinute prin ordonarea crescătoare a valorilor x_i, ordonare crescătoare care poate fi interpretată ca o permutare oarecare a indicilor valorilor ($\{1, 2, ..., n\} \longrightarrow \{i_1, i_2, ..., i_n\}$). În acest caz, evenimentul de interes exprimat de (5.6) este realizat prin mai multe evenimente elementare de tipul:

$$x_{i_1}, x_{i_2}, ..., x_{i_{k-1}} \leqslant t \leqslant x_{i_k} \leqslant t+dt \leqslant x_{i_{k+1}}, ..., x_{i_n} \quad (5.7)$$

Probabilitatea evenimentului elementar din (5.7) este dată de probabilitatea ca $k-1$ valori să fie mai mici decât t (deci $F^{k-1}(t)$), probabilitatea ca $n-k$ valori să fie mai mari

[1] Această ipoteză a mai fost făcută şi pentru deducerea transformării de egalizare a histogramei imaginilor.

ca $t + dt$ (deci $(1 - F(t))^{n-k}$) şi probabilitatea ca o valoare să fie cuprinsă în intervalul $[t; t + dt]$ (deci $f(t)dt$), adică:

$$P = F^{k-1}(t)(1 - F(t))^{n-k} f(t)dt$$

Numărul total de evenimente de tipul (5.7) este nC_{n-1}^{k-1} (n moduri de alege o valoare din cele n, la fiecare dintre aceste alegeri corespunzând C_{n-1}^{k-1} moduri de a alege din cele $n-1$ valori rămase $k-1$ valori care să fie mai mici ca t). Atunci:

$$\Pr\left(t \leqslant x_{(k)} \leqslant t + dt\right) = nC_{n-1}^{k-1} F^{k-1}(t)(1 - F(t))^{n-k} f(t)dt \tag{5.8}$$

şi deci funcţia de densitate de probabilitate a ieşirii filtrului de ordine de ordin k este:

$$f_k(x) = nC_{n-1}^{k-1} F^{k-1}(x)(1 - F(x))^{n-k} f(x) \tag{5.9}$$

În [12] sunt prezentate diferite utilizări ale funcţiei de densitate de probabilitate a ieşirii filtrelor de ordine, mai ales pentru determinarea comportării asimptotice ($n \longrightarrow \infty$) a filtrelor.

Proprietăţile deterministe ale filtrelor de ordine se referă la comportamentul acestora în zonele tipice ale unei imagini: porţiuni netede şi contururi. Filtrele de ordine nu afectează contururile (zonele de pantă abruptă) şi păstează valoarea zonelor uniforme. Aceste comportări au fost extinse la investigarea clasei mai generale de semnale rădăcină - semnale ce nu sunt modificate prin filtrare.

Orice secvenţă $\{x_i\}$ monotonă (crescătoare sau descrescătoare) este un semnal rădăcină pentru filtrele de ordine. În plus, filtrele de ordine comută cu orice funcţie monotonă g:

$$\text{rank}_k\left(g(x_i)\right) = g(\text{rank}_k(x_i))$$

Pe lângă metodele analitice de construire a semnalelor rădăcină [12], în practică, semnalele rădăcină se obţin prin filtrarea repetată a unui semnal oarecare, până la obţinerea invarianţei. Să considerăm spre exemplu secvenţa unidimensională $x = \{0, 1, 1, 3, 1, 3, 2, 3, 3, 2, 1, 1\}$ pe care o vom filtra cu o fereastră de filtrare de dimensiune 3, centrată, cu un filtru de ordine de ordin 2. După prima filtrare se obţine secvenţa $x1 = \{0, 1, 1, 1, 3, 2, 3, 3, 3, 2, 1, 1, 0\}$, iar după a doua filtrare se obţine secvenţa $x2 = \{0, 1, 1, 1, 2, 3, 3, 3, 3, 2, 1, 1, 0\}$. Această secvenţa este invariantă la filtrările următoare şi este deci un semnal rădăcină. Figura 5.4 prezintă secvenţele x, $x1$, $x2$.

5.1.1 Filtrul median

Filtrul median este un filtru de ordine a cărui ieşire este statistica de ordine de ordin central a setului de valori selectate de fereastra de filtrare.

$$\text{median}\{x_1, x_2, ..., x_n\} = \begin{cases} x_{\left(\frac{n+1}{2}\right)} \text{ dacă } n \text{ este impar} \\ \frac{1}{2}x_{\left(\frac{n}{2}\right)} + \frac{1}{2}x_{\left(\frac{n}{2}+1\right)} \text{ dacă } n \text{ este par} \end{cases} \tag{5.10}$$

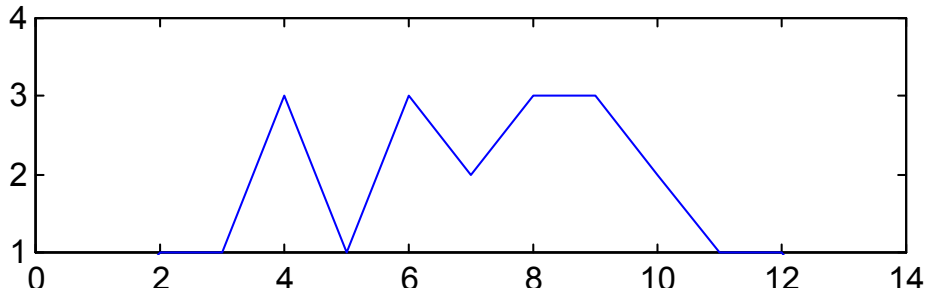

Fig. 5.2: Semnal iniţial oarecare; prin filtrări de ordine repetate acesta se va transforma într-un semnal rădăcină.

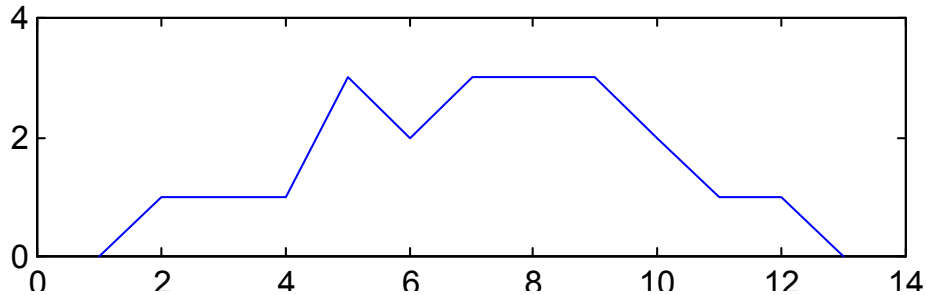

Fig. 5.3: Rezultatul primei filtrări a secvenţei iniţiale.

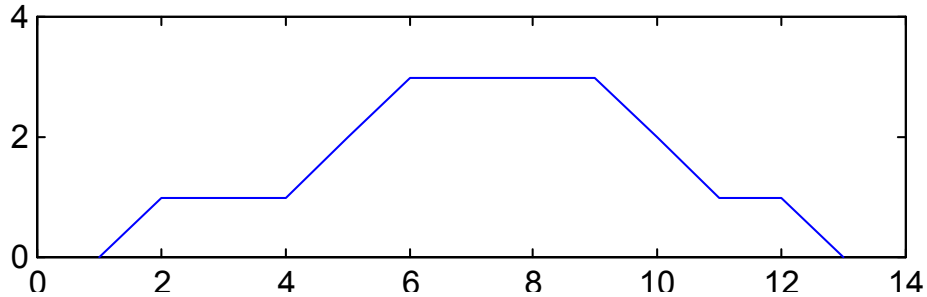

Fig. 5.4: Semnal rădăcină al filtrului de ordine considerat, obţinut după două iteraţii ale filtrării.

Ieşirea filtrului median este deci valoarea din centrul secvenţei ordonate; în cazul secvenţelor (ferestrelor de filtrare) de dimensiune pară, această poziţie centrală nu există, şi atunci este definită ca media aritmetică a valorilor vecine centrului imaginar. În practică, filtrul median se foloseşte doar cu ferestre de filtrare de dimensiune impară, ceea ce conduce la posibilitatea de a scrie:

$$\text{median}\{x_1, x_2, ..., x_n\} = \text{rank}_{\frac{n+1}{2}}\{x_1, x_2, ..., x_n\} \tag{5.11}$$

Aceasta înseamnă că dacă fereastra de filtrare are 3 puncte, medianul este statistica de ordinul 2[2], pentru o fereastră filtrare cu 9 puncte, medianul este statistica de ordinul 5, iar pentru o fereastră de filtrare de 25 de puncte, medianul este statistica de ordinul 13. Figura 5.5 prezintă rezultatul filtrării cu un filtru median cu fereastră pătrată de 3 x 3 puncte a imaginii degradate cu zgomot impulsiv din figura 5.1.

Fig. 5.5: Imagine filtrată cu filtru median cu fereastră de 3 x 3.

Efectele de filtrare a zgomotului impulsiv (de tip sare şi piper) sunt evidente; valorile punctelor de zgomot sunt 0 sau 255 şi deci, după ordonare se vor afla la "capetele" şirului de valori; ieşirea filtrului median, fiind statistica de ordin central, este situată departe de valorile extreme. În imaginea filtrată median (figura 5.5) se observă totuşi existenţa unor puncte albe şi negre (puncte de zgomot) ce nu au putut fi eliminate prin filtrare; spunem că în acele poziţii filtrul a fost străpuns de impulsuri (care deci au ajuns la ieşirea filtrului). Probabilitatea de străpungere a unui filtru median este dependentă de procentul de puncte de zgomot din imagine şi de dimensiunea ferestrei de filtrare folosite. Străpungerea filtrului median se produce atunci când în fereastra de filtrare avem măcar $\frac{n+1}{2}$ impulsuri de zgomot de aceeaşi valoare.

O variantă a filtrului median cu fereastră de filtrare pătrată este filtrul median separabil. Separabilitatea semnifică (ca şi în cazul transformărilor unitare) realizarea operaţiei întâi

[2]Aceasta înseamnă deci că exemplul de construire a semnalului rădăcină prezentat anterior este realizat pentru un filtru median.

pe linii, şi apoi pe coloanele imaginii, pe baza unor ferestre de filtrare liniare. Este evident că rezultatul acestei operaţii de filtrare nu coincide cu rezultatul filtrării mediane cu fereastră pătrată[3] (figura 5.6 prezintă rezultatul filtrării mediane separabile cu ferestre de dimensiune 3 a imaginii cu zgomot impulsiv din figura 5.1; rezultatul filtrării mediane cu fereastră pătrată 3 x 3 este prezentat în figura 5.5).

Fig. 5.6: Rezultatul filtrării cu un filtru median separabil.

Se remarcă faptul că această structură de filtrare, deşi prezintă avantaje din punctul de vedere al timpului de calcul necesar, se străpunge mai uşor decât filtrul median cu fereastră pătrată din care a derivat.

5.1.2 Filtrele de ordine ponderate şi structurile multietaj

Atât filtrele liniare cât şi filtrele de ordine se bazează pe acelaşi principiu al ferestrei glisante. Prelucrările realizate asupra valorilor selectate de această fereastră de filtrare sunt evident diferite, dar se poate totuşi remarca faptul că structura de filtrare liniară permite o flexibilitate mai mare (se pot realiza un număr infinit de filtre liniare pentru o aceeaşi fereastră de filtrare, prin varierea coeficienţilor de ponderare). Singurul factor de reglaj al filtrelor de ordine este ordinul statisticii selectate la ieşire[4].

Este evident că ponderarea filtrelor de ordine nu se poate face prin multiplicarea valorilor selectate de fereastra de filtrare cu diferiţi coeficienţi, aşa ca în cazul filtrelor liniare.

[3] Filtrul de minim (filtrul de ordine de ordin 1) şi filtrul de maxim (filtrul de ordine de ordin n) sunt singurele filtre de ordine separabile.

[4] Zamperoni a propus adaptarea ordinului statisticii folosite ca ieşire a filtrului de ordine prin $k = \left[\frac{1}{2} + \sum_{i=1}^{n} \frac{x_{(i)} - x_{(1)}}{x_{(n)} - x_{(1)}}\right]$.

Ponderarea are ca scop întărirea influenţei anumitor pixeli din fereastra de filtrare (de exemplu pixelul central - curent prelucrat - care ar fi de aşteptat ca să aibă o influenţă mai puternică asupra rezultatului filtrării decât vecinii săi). Având în vedere că rezultatul filtrării (deci valoarea unei anumite statistici) este determinată în urma ordonării, modalitatea de a întări influenţa unei anume valori este de a o repeta de mai multe ori, crescând astfel probabilitatea de a o regăsi ca statistica de ordin dorit. Aşadar, ponderarea filtrelor de ordine se referă la repetarea de un număr fixat de ori a valorilor selectate de fereastra de filtrare. Mulţimea de valori ce rezultă se numeşte multiset. Ponderile w_i ataşate fiecărei poziţii de filtrare vor fi numere naturale nenule, ce exprimă numărul de repetiţii al valorii corespunzătoare în multiset.

Spre exemplu să considerăm porţiunea de imagine selectată de o fereastră de filtrare de dimensiune 3 x 3, ale cărei valori sunt $\begin{pmatrix} 1 & 3 & 3 \\ 2 & 2 & 1 \\ 4 & 3 & 5 \end{pmatrix}$; statistica mediană ce corespunde acestei situaţii este 3 (setul ordonat de valori fiind 1, 1, 2, 2, 3, 3, 3, 4, 5). Dacă considerăm acum filtrul median ponderat cu masca $W = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 3 & 2 \\ 1 & 2 & 1 \end{pmatrix}$, multisetul rezultant este 1, 3, 3, 3, 2, 2, 2, 2, 2, 1, 1, 4, 3, 3, 5 (sau ordonat 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 4, 5) ceea ce face ca medianul să fie 2 (şi deci o valoare mai apropiată de valoarea originală din punctul curent prelucrat). Masca de ponderare W semnifică faptul că valoarea centrală este repetată de 3 ori, valorile de pe poziţiile vecinilor imediaţi (orizontali şi verticali) sunt repetate de câte 2 ori, iar valorile de pe poziţiile vecinilor de pe diagonală sunt considerate o singură dată. Filtrul de ordine rezultat se poate scrie ca:

$$\text{rank}_k(x_i \diamond w_i) = x_{(k)}$$

Este evident că multisetul de valori conţine $\sum_{j=1}^{n} w_i$ termeni, şi deci ordinul statisticii maxime este $\sum_{j=1}^{n} w_i$, iar ordinul statisticii mediane este $\frac{1}{2}\left(\sum_{j=1}^{n} w_i + 1\right)$ (presupunând un număr impar de valori în multiset). Un caz particular destul de des utilizat este acela al filtrelor de ordine central ponderate, în care toţi coeficienţii de ponderare (repetiţie) sunt 1, cu excepţia coeficientului ce corespunde pixelului curent prelucrat, ce are o valoare mai mare ca 1.

Structurile multietaj grupează câteva filtre de ordine (median, minim, maxim) aplicate succesiv, cu ferestre de filtrare diferite. Cele mai răspândite structuri sunt cele de tip median multietaj (median din median pe orizontală, median pe verticală şi valoarea curentă), reprezentat în figura 5.8 şi max/min-median (maxim sau minim din medianele pe verticală, orizontală şi cele două diagonale ale ferestrei de filtrare pătrate centrate în punctul curent prelucrat), reprezentat în figura 5.7.

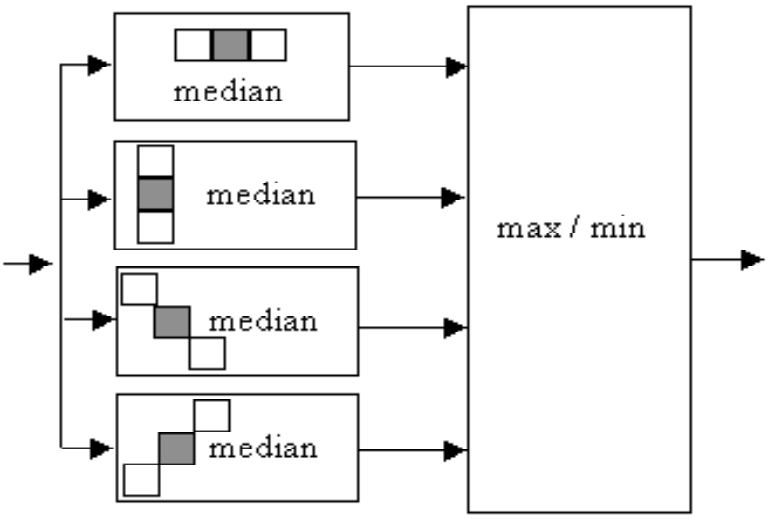

Fig. 5.7: Structră multietaj de tip max / min - median, bazată pe o fereastră de filtrare de dimensiune 3.

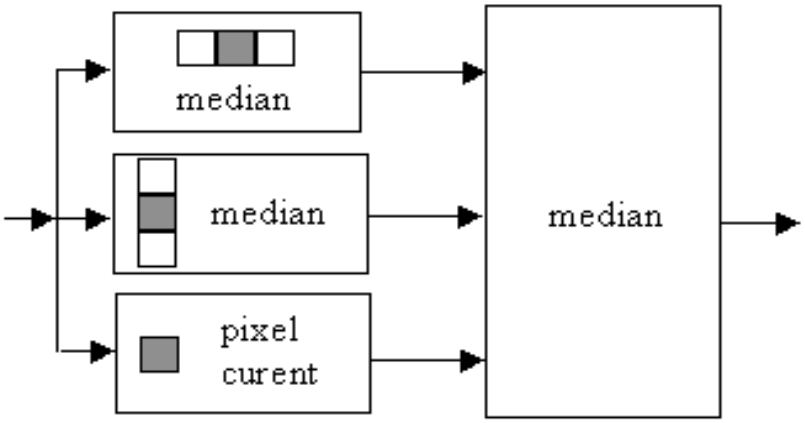

Fig. 5.8: Structură de filtrare de tip median - median, bazată pe o fereastră de filtrare pătrată de dimensiune 3.

5.2 Filtre de ordine de domeniu

Filtrele de ordine de domeniu (filtre LUM – *Lower-Upper-Middle* [18]) introduc o posibilitate de adaptare şi reglare a filtrelor de ordine.

Valoarea y de ieşire a filtrului LUM de netezire este definită ca:

$$y = \begin{cases} x_{(k)}, \text{ dacă } x^* < x_{(k)} \\ x_{(n-k+1)}, \text{ dacă } x^* > x_{(n-k+1)} \\ x^*, \text{ în rest} \end{cases} \tag{5.12}$$

unde x^* este valoarea ce corespunde originii ferestrei de filtrare cu n poziţii, iar k este parametrul de reglaj al filtrului. Valoarea k este un întreg din intervalul $[0; \frac{n+1}{2}]$; dacă $k = 0$ filtrul de comportă ca un filtru trece tot ($y = x^*$), iar dacă $k = \frac{n+1}{2}$, filtrul este un filtru median ($y = x_{(\frac{n+1}{2})}$).

Valoarea y de ieşire a filtrului LUM de conturare este definită ca:

$$y = \begin{cases} x_{(l)}, \text{ dacă } x_{(l)} < x^* < \frac{x_{(l)}+x_{(n-l+1)}}{2} \\ x_{(n-l+1)}, \text{ dacă } \frac{x_{(l)}+x_{(n-l+1)}}{2} < x^* < x_{(n-l+1)} \\ x^*, \text{ în rest} \end{cases} \tag{5.13}$$

unde x^* este valoarea ce corespunde originii ferestrei de filtrare cu n poziţii, iar l este parametrul de reglaj al filtrului. Valoarea k este un întreg din intervalul $[0; \frac{n+1}{2}]$; dacă $l = 0$ filtrul de comportă ca un filtru de reliefare extremă, iar dacă $l = \frac{n+1}{2}$, filtrul este un filtru trece tot ($y = x^*$).

5.3 L-filtre

L-filtrele [12] au fost introduse din dorinţa de a oferi o mai mare flexibilitate operaţiilor de filtrare neliniară bazate pe ordonare; pasul firesc de dezvoltare a fost combinarea tuturor statisticilor de ordine în valoarea de la ieşirea filtrului. Ieşirea unui L-filtru este o combinaţie liniară a statisticilor de ordine:

$$L(\{x_1, x_2, ..., x_n\}) = \sum_{i=1}^{n} w_i x_{(i)} \tag{5.14}$$

Este evident că prin varierea coeficienţilor w_i se pot obţine diferite filtre, inclusiv filtrele de ordine (un singur coeficient egal cu 1, iar restul nuli). Coeficienţii celor mai uzuale L-filtre sunt prezentaţi în tabelul 5.1

Filtru	Coeficienţi
Filtru de ordine de ordin k, rank_k	$a_k = 1$, $a_i = 0$, $i \neq k$
Filtru de mediere aritmetică	$a_i = 1/n$
Filtru de mijloc	$a_1 = a_n = 0.5$, $a_i = 0$, $i \notin \{1, n\}$
Filtru de cvasi-mijloc	$a_k = a_{n+1-k} = 0.5$, $a_i = 0$, $i \notin \{k, n+1-k\}$
Medie α-reglabilă	$a_k = 1/n(1-2\alpha)$, $i \in [\alpha n + 1; n - \alpha n]$ $a_i = 0$, $i \notin [\alpha n + 1; n - \alpha n]$

Tabel 5.1: Coeficienţii celor mai uzuale L-filtre.

Se poate demonstra cu uşurinţă că coeficienţii L-filtrelor trebuie să îndeplinească aceleaşi condiţii de normare ca şi coeficienţii filtrelor liniare: să aibă suma 1 pentru un filtru de netezire (conform (3.8)) şi suma 0 pentru un filtru de reliefare (conform (3.9)). Toate filtrele prezentate în tabelul 5.1 sunt deci filtre de netezire, al căror scop este mărirea uniformităţii regiunilor imaginii prin eliminarea diferitelor zgomote suprapuse acesteia. Tabelul 5.2 prezintă filtrele cele mai potrivite pentru eliminarea diferitelor tipuri de zgomote.

Tip zgomot	Filtru
impulsiv (sare şi piper)	filtru de ordine (median, maxim, minim)
gaussian	filtru de mediere aritmetică
impulsiv + gaussian	filtru de tip medie α-reglabilă
uniform	filtru de mijloc

Tabel 5.2: L-filtre utilizate pentru reducerea diferitelor tipuri de zgomote.

Un exemplu de filtru de reliefare (cu comportare derivativă, care să satisfacă deci (3.9)) este gradientul morfologic – diferenţa între valoarea maximă şi minimă din fereastra de filtrare curentă:

$$grad\{x_1, x_2, ..., x_n\} = x_{(n)} - x_{(1)}$$

Folosirea unui asemenea operator produce o imagine în care valoarea fiecărui pixel este invers proporţională cu gradul de uniformitate (netezime) al vecinătăţii acestuia (vezi figura 5.9).

5.4 Aspecte de implementare

Implementarea filtrelor neliniare de ordine sau a L-filtrelor urmează aceleaşi principii ca şi oricare tehnică de tip fereastră glisantă (ca filtrarea liniară) în ceea ce priveşte

Fig. 5.9: Operatorul gradient morfologic aplicat imaginii filtrate din figura 5.5.

problemele legate de realizarea formei ferestrei de filtrare şi efectele de margine. Ceea ce este particular filtrelor din această categorie este etapa de ordonare a valorilor selectate de fereastra de filtrare în fiecare poziţie.

În teoria algoritmilor au fost dezvoltaţi numeroşi algoritmi de sortare (ordonare crescătoare), a căror complexitate de calcul variază de la $O(n^2)$ (bubble sort, sortare prin determinare succesivă de extreme), $O(n \log n)$ (sortare prin interclasare sau inserţie) până la $O(n)$ (pentru determinare doar a extremelor şi a statisticii mediane). Trebuie însă remarcat că o metodă de sortare de complexitate $O(n^2)$ nu este neapărat mai neeficientă decât o metodă de sortare $O(n)$: complexitatea unui algoritm exprimă comportarea asimptotică a numărului de operaţii necesare, în condiţiile în care dimensiunea caracteristică n a setului de date ce se prelucrează este foarte mare (la limită $n \longrightarrow \infty$). Sortările folosite în implementarea filtrelor neliniare de ordine prelucrează seturi mici de date (să nu uităm că rareori o fereastră de filtrare depăşeşte dimensiunea de 25 de pixeli), şi deci complexitatea $O()$ nu este un criteriu valid de alegere a unui algoritm de sortare.

Codul Matlab ce urmează implementează un L-filtru cu o fereastră pătrată de dimensiune 3 x 3, centrată; ponderile filtrului sunt stocate în vectorul w (1 linie, 9 coloane, prima poziţie corespunzând coeficientului statisticii minime). L-filtrul este aplicat imaginii *in* (de dimensiuni *nrlin* şi *nrcol*); efectele de margine sunt evitate prin neprelucrarea primei şi ultimei linii, respectiv coloane a imaginii. Rezultatul este înscris în imaginea *out*.

```
[nrlin,nrcol]=size(in);
out=in;
for i=2:nrlin-1
 for j=2:nrcol-1
  temp=in(i-1:i+1,j-1:j+1);
  temp=sort(temp(:));
```

```
  out(i,j)=fix(w*temp);
 end
end
```

Se remarcă folosirea funcţiei standard *sort* ce realizează operaţia de ordonare crescătoare a valorilor; ieşirea L-filtrului este produsul scalar al vectorului de coeficienţi ai filtrului cu vectorul statisticilor de ordine.

În cele ce urmează vom considera un exemplu de implementare a unui filtru cu ordonare după rang relizat cu o fereastră centrată de 3 x 3 pixeli, pentru filtrarea unei imagini cu *NRLIN* linii şi *NRCOL* coloane; prima şi ultima linie şi coloană sunt identice cu cele din imaginea ce se filtrează (remarcaţi buclele *for* de alocare a spaţiului pentru imaginea filtrată şi iniţializarea valorilor acesteia).

```
unsigned char temp[9],dummy;
boolean sortat;
img_out=(int**)malloc(NRLIN*sizeof(int*));
for (i=0;i<NRLIN;i++)
  img_out[i]=(int*)malloc(NRCOL*sizeof(int));
for (i=0;i<NRCOL;i++)
{ img_out[0][i]=img[0][i];
  img_out[NRLIN-1][i]=img[NRLIN-1][i];}
for (i=0;i<NRLIN;i++)
{ img_out[i][0]=img[i][0];
  img_out[i][NRCOL-1]=img[i][NRCOL-1];}
for (i=1;i<NRLIN-1;i++)
 for (j=1;j<NRCOL-1;j++)
 { for (k=0;k<9;k++)
    temp[k]=img[i-1+k/3][j-1+k%3];
   sortat=false;
   while (! sortat) do
   {  sortat=true;
      for (k=0;k<8;k++)
      if (temp[k]>temp[k+1])then
       { dummy=temp[k+1];
         temp[k+1]=temp[k];
         temp[k]=dummy;
         sortat=false;
        }
    }
   img_out[i][j]=temp[5];
}
```

Valorile selectate de fereastra de filtrare sunt stocate în vectorul de 9 poziţii *temp*. Citirea acestor valori se face printr-un ciclu, bazându-se pe introducerea unei relaţii de echivalenţă între ordinea de baleiaj a ferestrei şi poziţia valorilor în vector (economia nu este de timp de execuţie ci de timp de scriere a codului). Această alocare este echivalentă cu liniile de cod:

```
temp[0]=img[i-1][j-1];temp[1]=img[i-1][j];
temp[2]=img[i-1][j+1];temp[3]=img[i][j-1];
temp[4]=img[i][j];temp[5]=img[i][j+1];
temp[6]=img[i+1][j-1];temp[7]=img[i+1][j];
temp[8]=img[i+1][j+1];
```

Sortarea implementată este de tip bubble sort; fiecare poziţie a vectorului ce conţine valorile extrase de fereastra de filtrare este comparată cu poziţia următoare, iar dacă ordinea crescătoare nu este respectată, cele două valori sunt interschimbate; indicatorul *sortat* indică efectuarea a măcar unei interschimbări, şi deci necesitatea reluării verificării ordinii. Filtrul de ordine folosit este medianul (se preia în imaginea rezultat statistica de ordine cu numărul 5, deci statistica mediană).

Structura de program folosită anterior realizează ordonarea vectorului de valori pentru fiecare poziţie a ferestrei de filtrare; o serioasă economie de timp se poate realiza dacă se ţine seama că la mutarea ferestrei de filtrare de la un pixel al imaginii la un pixel vecin acestuia, se modifică un număr relativ mic de valori (restul rămânând neschimbate). Aceasta poate conduce la ideea păstrării unei părţi a setului de valori ordonate, în care să se intercaleze valorile noi.

O altă variantă de determinare a statisticilor de ordine se bazează pe folosirea histogramei valorilor din fereastra de filtrare. Histograma (2.10) reprezintă numărul de puncte ce au o anumită valoare şi este echivalentă cu funcţia de densitate de probabilitate a unei variabile aleatoare. Pentru aceeaşi varibilă aleatoare există însă şi funcţia de repartiţie – primitiva funcţiei de densitate de probabilitate; aşadar putem asocia oricărei histograme h o histogramă cumulativă (5.15):

$$H(i) = \sum_{j=0}^{i} h(j), \text{ cu } i = 0, 1, ..., L-1 \tag{5.15}$$

Histograma cumulativă în punctul i va reprezenta deci numărul de puncte (pixeli) a căror valoare (nivel de gri) este mai mică decât i. Determinarea valorii statisticilor de ordine pentru un set de valori pe baza histogramei acestora este imediată. Să considerăm următorul exemplu: setul de valori selectate de fereastra de filtrare este {1, 2, 3, 3, 4, 0, 1, 1, 2, 1, 1, 0, 0, 1, 2, 7, 7, 6, 0, 1, 6, 6, 5, 5, 0}(valori din intervalul 0–7). Histograma acestui set este $h = \begin{pmatrix} 5 & 7 & 3 & 2 & 1 & 2 & 3 & 2 \end{pmatrix}$ iar histograma cumulativă este $H = \begin{pmatrix} 5 & 12 & 15 & 17 & 18 & 20 & 23 & 25 \end{pmatrix}$. Atunci statisticile de ordine de ordin 1–5

($H(0)$) au valoarea 0, statisticile de ordine de ordin 6 ($H(0)+1$) – 12 ($H(1)$) au valoarea 1, statisticile de ordine de ordin 13 ($H(1)+1$) – 15 ($H(2)$) au valoarea 2, statisticile de ordine de ordin 16 ($H(2)+1$) – 17 ($H(3)$) au valoarea 3, statistica de ordine de ordin 18 ($H(3)+1 = H(4)$) are valoarea 4, statisticile de ordine de ordin 19 ($H(4)+1$) – 20 ($H(5)$) au valoarea 5, statisticile de ordine de ordin 21 ($H(5)+1$) – 23 ($H(6)$) au valoarea 6, statisticile de ordine de ordin 24 ($H(6)+1$) – 25 ($H(7)$) au valoarea 7. Regula de decizie generală este: statisticile de ordine de ordin $H(j-1)+1$ până la $H(j)$ au valoarea j, cu $j = 0, 1, 2, ..., L-1$ şi $H(-1) = 0$.

Deci, pentru o statistică oarecare de ordin r, trebuie identificat numărul j astfel ca:

$$H(j-1)+1 \leqslant r \leqslant H(j) \tag{5.16}$$

Problemele ce apar la o asemenea implementare sunt legate în primul rând de timpul de determinare a numărului j ce satisface (5.16), timp ce este direct proporţional cu numărul de nivele de gri din imagine (şi deci lungimea vectorului histogramă sau histogramă cumulativă). Se poate însă observa că histograma într-o fereastră de filtrarea va selecta relativ puţine valori diferite, şi atunci histograma va fi un vector rar (*sparse*), cu multe intrări nule, iar histograma cumulativă va avea, corespunzător, multe porţiuni constante. În aceste condiţii, o reprezentare mai eficientă (atât ca memorie ocupată, cât şi ca timp de căutare) este memorarea doar a tranziţiilor (poziţie, valoare) din histograma cumulativă, adică doar memorarea histogramei.

Capitolul 6

ELEMENTE DE MORFOLOGIE MATEMATICĂ

În mod tradiţional, prelucrarea semnalelor multidimensionale (şi a imaginilor în particular) a fost bazată pe exploatarea conceptelor şi teoriei sistemelor liniare şi a transformatei Fourier ([9], [2]). Deşi aceste abordări clasice sunt justificate şi dau rezultate în multe cazuri, aplicarea lor este limitată de problema fundamentală impusă de semnalele de tip imagine: modul de reprezentare a formei sau structurii geometrice existente într-un semnal.

Morfologia matematică, dupa cum indică şi numele (*morphos* – formă, *logos* – ştiinţă, deci ştiinţa formelor), realizează o abordare axată pe formă a prelucrării imaginilor. Folosită corespunzător, morfologia matematică conduce la prelucrări ce simplifică structura imaginii, păstrând caracteristicile esenţiale de formă şi eliminând irelevanţele.

Ideea de bază a oricărei prelucrări morfologice este considerarea imaginii ca un ansamblu (mulţime, reuniune de părţi) asupra căruia se aplică transformări a căror esenţă este comparaţia cu mulţimi (ansambluri) mai simple, numite *elemente structurante*. Scopul acestor transformări este extragerea de forme mai simple din formele iniţiale (complexe) ale imaginii.

Morfologia matematică este utilizată ca o abordare naturală a proceselor de vedere artificială, deoarece trăsăturile şi respectiv identificarea obiectelor sunt corelate cu forma. Principalele aplicaţii sunt în domeniile roboticii, microscopiei electronice, imagisticii biomedicale, telemetriei, inspecţiei automate a produselor, analizei de scenă. Aplicaţiile industriale sunt impulsionate şi de continua dezvoltare şi îmbunatatire a arhitecturilor de calcul ce implementează transformări morfologice.

6.1 Transformări morfologice de bază

6.1.1 Transformarea Hit or Miss

Transformarea *Hit or Miss* a fost introdusă în [14] şi reprezintă poate cea mai primară şi evidentă exemplificare a conceptului de studiu al formei. Orice formă poate fi considerată ca o reuniune de componente (blocuri, regiuni, scheme) individuale independente, plasate în diverse poziţii; a recunoaşte forma implică identificarea locală a părţilor sale componente şi deci o operaţie simplă de potrivire de măşti (*pattern matching*).

Rezultatul aplicării acestei transformări de identificare (numită uneori şi transformarea "totul sau nimic") este o mulţime ale cărei puncte satisfac criteriul de identificare a unei vecinătăţi cu masca aplicată.

Transformarea *Hit or Miss* se defineşte pe baza unei partiţii $(B_1,\ B_2)$ a elementului structurant B: $B = B_1 \cup B_2$ şi $B_1 \cap B_2 = \emptyset$ ca:

$$A \circledast B = \{\mathbf{x}|(B_1)_\mathbf{x} \subseteq A\} \cap \{\mathbf{x}|(B_2)_\mathbf{x} \subseteq A^C\} \tag{6.1}$$

În (6.1) A este mulţimea căreia i se aplică transformarea, A^C este complementara mulţimii A, B este elementul structurant şi $(B_i)_\mathbf{x}$ este mulţimea B_i translatată cu vectorul $\mathbf{x}$. Trebuie remarcat faptul că oricărui element structurant trebuie să i se ataşeze o origine. Figura 6.1 prezintă o exemplificare a transformării *Hit or Miss*.

Se poate observa că această transformare produce ca rezultat o mulţime de puncte ce satisfac concomitent un set de condiţii de tipul $(B_i)_\mathbf{x} \subseteq A_i$, unde $\{A_i\}$ formează o partiţie a spaţiului.

Este evident că $\{A, A^C\}$ formează o partiţie, dar aceasta nu este singura posibilă. Anumite aplicaţii practice pot impune însă situaţii mai complexe, a căror rezolvare depăşeşte cadrul morfologiei binare [14]. Astfel se pot cita aşa numitele "cazuri": cazul petrografic (provenit din analiza hidrocarburilor şi a mineralelor), în care p componente împart exact spaţiul (fără a lăsa locuri libere) şi cazul geografic, în care p componente nu ocupă întreg spaţiul şi se pot întrepătrunde (cazul zonelor de pădure cu diferite specii de copaci). Figura 6.2 prezintă o astfel de transformare.

Transformarea *Hit or Miss*, în forma prezentată, prezintă un interes mai mult teoretic, datorită situării sale la baza construcţiei morfologiei matematice. Se va arăta însă că este posibilă exprimarea acesteia şi în funcţie de transformările morfologice fundamentale larg utilizate (erodarea şi dilatarea).

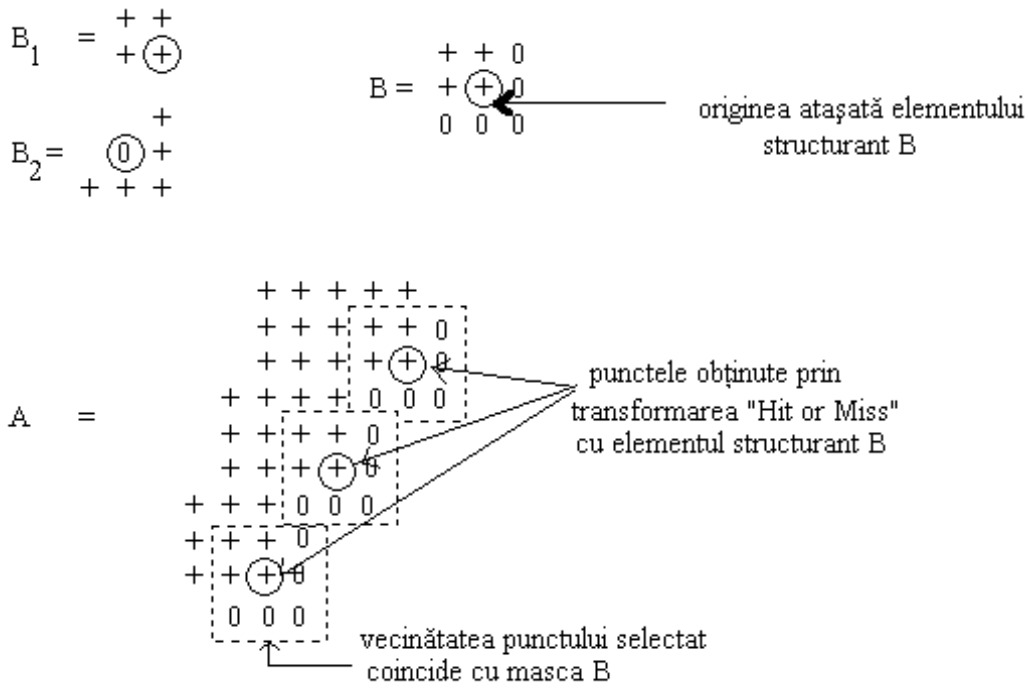

Fig. 6.1: Exemplificare a transformării de identificare a configuraţiilor locale (Hit or Miss); în particular exemplul prezintă determinarea punctelor de tip "colţ dreapta-jos".

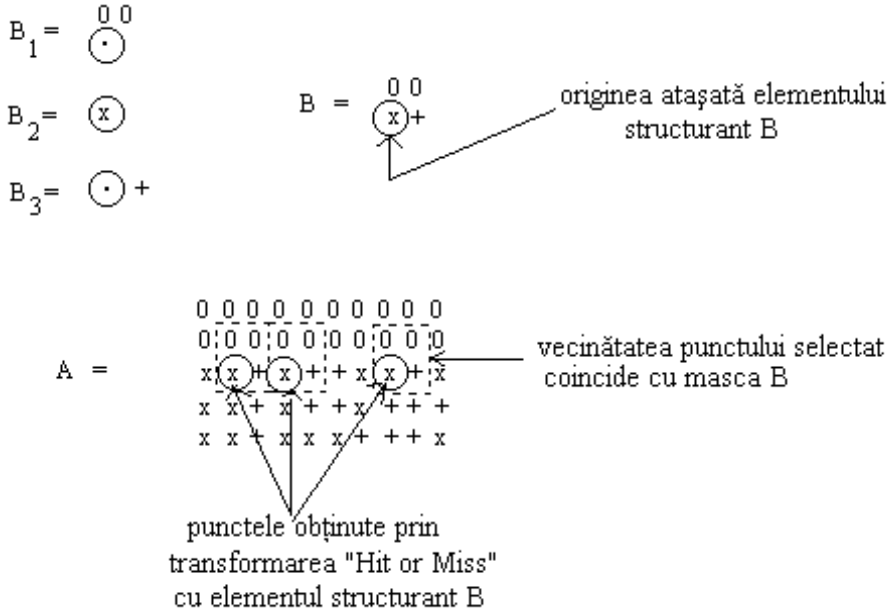

Fig. 6.2: Exemplu de transformare Hit or Miss extinsă pentru o partiţie cu trei mulţimi a spaţiului imaginii.

6.1.2 Erodarea morfologică

Erodarea morfologică a mulţimii A prin elementul structurant B se defineşte ca mulţimea punctelor (elementelor) cu care se poate translata elementul structurant astfel încât acesta să fie inclus în mulţimea de prelucrat A:

$$A \ominus B = \{\mathbf{x} | B_{\mathbf{x}} \subseteq A\} \tag{6.2}$$

Erodare morfologică a mulţimii A este transformata Hit or Miss a mulţimii cu un element structurant $B = B_1$ ($B_2 = \emptyset$):

$$A \ominus B = \{\mathbf{x} | (B_1)_{\mathbf{x}} \subseteq A\} \cap \{\mathbf{x} | (\emptyset)_{\mathbf{x}} \subseteq A^C\} = \{\mathbf{x} | (B)_{\mathbf{x}} \subseteq A\}$$

Această relaţie de definiţie se mai poate exprima ca:

$$\begin{aligned} A \ominus B &= \{\mathbf{x} | B_{\mathbf{x}} \subseteq A\} = \{\mathbf{x} | \forall \mathbf{b} \in B, \exists \mathbf{a} \in A \text{ astfel încât } \mathbf{b} + \mathbf{x} = \mathbf{a}\} = & (6.3) \\ &= \{\mathbf{x} | \forall \mathbf{b} \in B, \exists \mathbf{a} \in A \text{ astfel încât } \mathbf{x} = \mathbf{a} - \mathbf{b}\} = \bigcap_{\mathbf{b} \in B} A_{-\mathbf{b}} = \bigcap_{\mathbf{b} \in B^S} A_{\mathbf{b}} & (6.4) \end{aligned}$$

Figura 6.3 prezintă rezultatul erodării unor mulţimi discrete, iar figura 6.4 prezintă rezultatul erodării unor mulţimi continue.

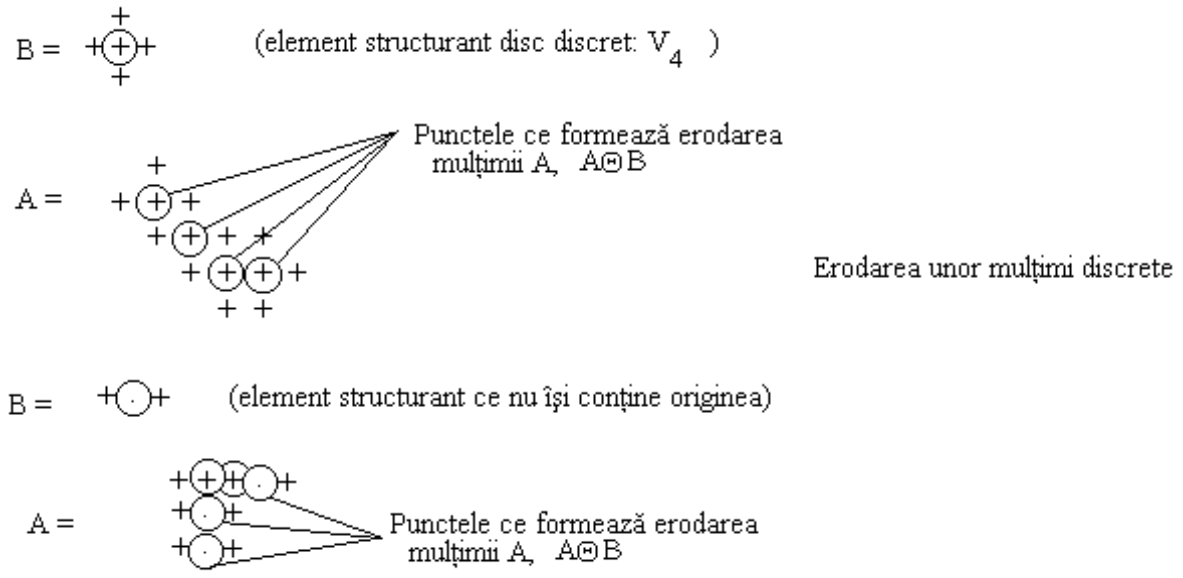

Fig. 6.3: Exemple de erodare a unor mulţimi discrete cu diferite elemente structurante, ce îşi conţin (sau nu) originea.

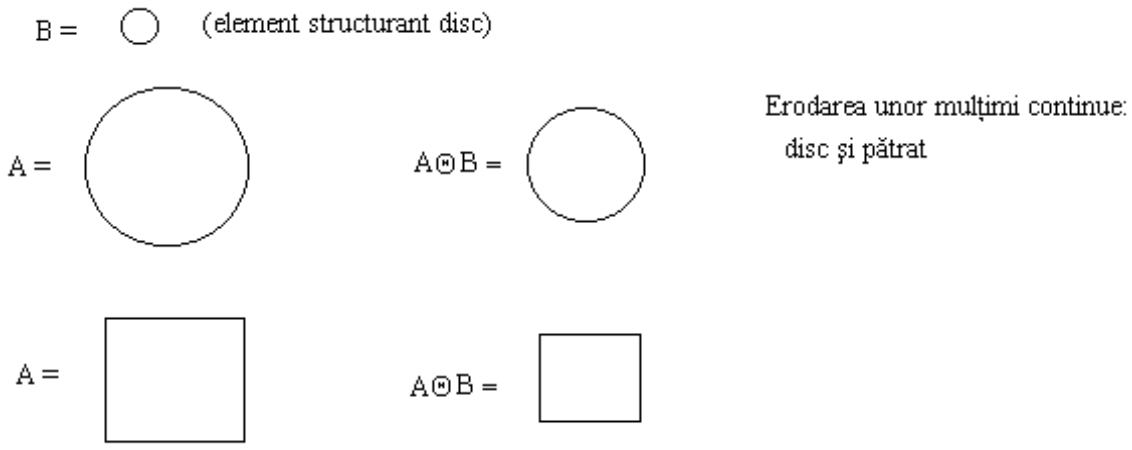

Fig. 6.4: Rezultatul erodării unor mulţimi continue cu un element structurant de tip disc (originea elementului structurant este centrul discului).

6.1.3 Dilatarea morfologică

Dilatarea morfologică a mulţimii A cu elementul structurant B se defineşte ca mulţimea punctelor (elementelor) cu care se poate translata elementul structurant astfel încât acesta să aibă puncte comune cu mulţimea de prelucrat A:

$$A \oplus B = \{\mathbf{x} | B_{\mathbf{x}} \cap A \neq \emptyset\} \tag{6.5}$$

Erodare morfologică a mulţimii A este complementara transformatei Hit or Miss a mulţimii cu un element structurant $B = B_2$ ($B_1 = \emptyset$):

$$(A \oplus B)^C = \{\mathbf{x} | (B_1)_{\mathbf{x}} \subseteq A\} \cap \{\mathbf{x} | (B_2)_{\mathbf{x}} \subseteq A^C\} = \{\mathbf{x} | (\emptyset)_{\mathbf{x}} \subseteq A\} \cap \{\mathbf{x} | (B_2)_{\mathbf{x}} \subseteq A^C\} = \{\mathbf{x} | (B)_{\mathbf{x}} \subseteq A^C\}$$

$$A \oplus B = \{\mathbf{x} | (B)_{\mathbf{x}} \subseteq A^C\}^C = \{\mathbf{x} | (B)_{\mathbf{x}} \nsubseteq A^C\} = \{\mathbf{x} | B_{\mathbf{x}} \cap A \neq \emptyset\}$$

Relaţia de definiţie mai poate fi exprimată şi ca:

$$A \oplus B = \{\mathbf{x} | B_{\mathbf{x}} \cap A \neq \emptyset\} = \{\mathbf{x} | \exists \mathbf{b} \in B, \exists \mathbf{a} \in A \text{ astfel încât } \mathbf{b} + \mathbf{x} = \mathbf{a}\} = \tag{6.6}$$

$$= \{\mathbf{x} | \exists \mathbf{b} \in B, \exists \mathbf{a} \in A \text{ astfel încât } \mathbf{x} = \mathbf{a} - \mathbf{b}\} = \bigcup_{\mathbf{b} \in B} A_{-\mathbf{b}} = \bigcup_{\mathbf{b} \in B^S} A_{\mathbf{b}} \tag{6.7}$$

Figura 6.5 prezintă rezultatul erodării unor mulţimi discrete, iar figura 6.6 prezintă rezultatul dilatării unor mulţimi continue.

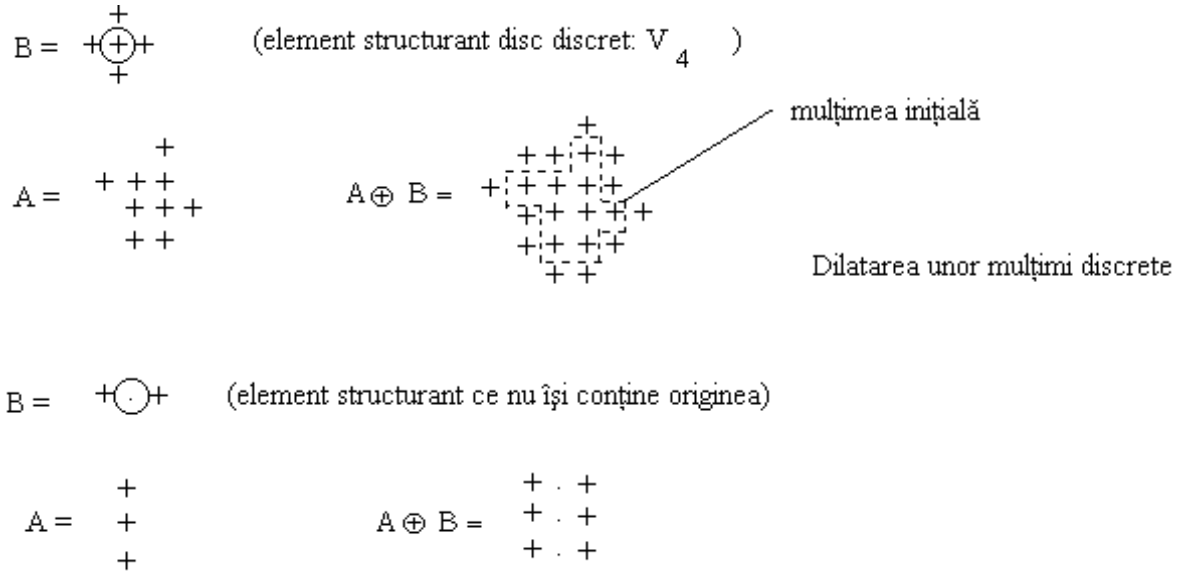

Fig. 6.5: Exemple de dilatare a unor mulţimi discrete cu diferite elemente structurante, ce îşi conţin (sau nu) originea.

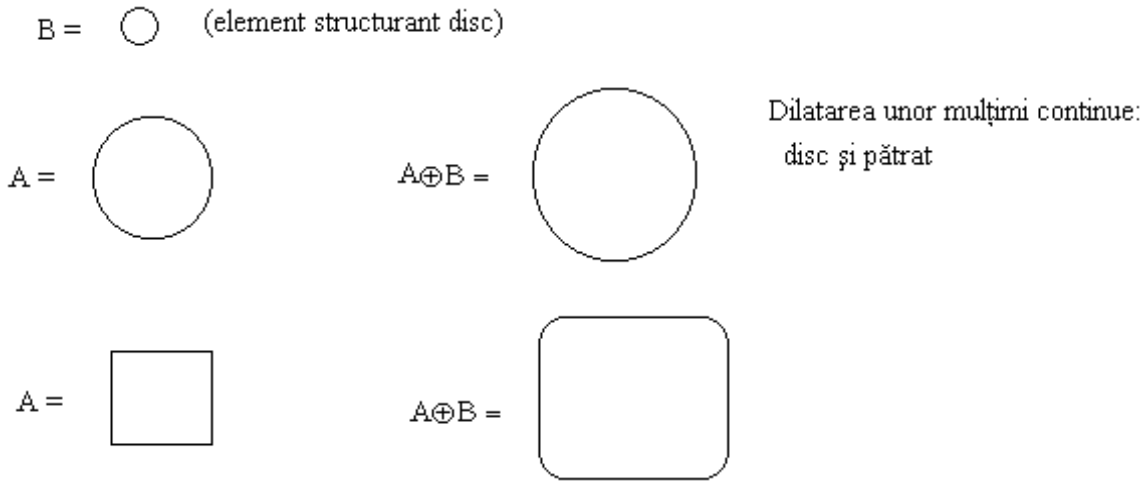

Fig. 6.6: Rezultatul dilatării unor mulţimi continue cu un element structurant de tip disc (originea elementului structurant este centrul discului).

6.1.4 Proprietăţile erodării şi dilatării

Se observă că, în general, efectul operaţiei de dilatare este de a mări obiectul; acesta creşte, "se umflă", corespunzător formei şi dimensiunii elementului structurant. Efectul erodării este, după cum am văzut, o micşorare a obiectului. Modificarea dimensiunii obiectului este strict determinată de forma elementului structurant: un element structurant simetric (disc, segment de dreaptă centrat în origine) provoacă o modificare simetrică a dimensiunilor; dacă elementul structurant nu este simetric, efectele se vor manifesta asupra obiectului pe direcţia elementului structurant şi în sens contrar acestuia. Efectele erodării şi dilatării vor fi discutate în continuare, pe baza proprietăţilor matematice ale acestora.

Dilatarea şi erodarea formează o pereche de operaţii duale:

$$(A \oplus B)^C = A^C \ominus B \text{ şi } (A \ominus B)^C = A^C \oplus B \tag{6.8}$$

Demonstraţia este imediată:

$A \oplus B = \{\mathbf{x} | B_\mathbf{x} \cap A \neq \emptyset\}$, de unde rezultă

$$(A^C \oplus B)^C = \{\mathbf{x} | B_\mathbf{x} \cap A^C \neq \emptyset\}^C = \{\mathbf{x} | B_\mathbf{x} \cap A^C = \emptyset\} = \{\mathbf{x} | B_\mathbf{x} \subseteq A\} = A \ominus B \tag{6.9}$$

şi $A \ominus B = \{\mathbf{x} | B_\mathbf{x} \subseteq A\}$ de unde rezultă

$$(A^C \ominus B)^C = \{\mathbf{x} | B_\mathbf{x} \subseteq A^C\}^C = \{\mathbf{x} | B_\mathbf{x} \nsubseteq A^C\} = \{\mathbf{x} | B_\mathbf{x} \cap A \neq \emptyset\} = A \oplus B \tag{6.10}$$

Interpretarea dualităţii ca obţinerea aceloraşi efecte, dar asupra unor mulţimi complementare, este corectă: dacă dilatarea va mări obiectul (mulţimea), aceasta înseamnă că va micşora în acelaşi timp fundalul (complementara mulţimii), deci va fi echivalentă cu o erodare a fundalului. Dacă erodarea micşorează obiectul (mulţimea), aceasta înseamnă o mărire simultană a fundalului (complementara mulţimii), deci o dilatare a acestuia.

Un caz particular ce poate apare este acela al elementelor structurante ce nu conţin originea. Folosind un asemenea element structurant, rezultatele operaţiilor morfologice vor fi "translatate" faţă de poziţia la care ar fi fost de aşteptat să apară.

Legătura între translaţia elementului structurant (faptul că acesta nu conţine originea poate fi interpretat ca o deplasare) sau a mulţimii (obiectului) care se prelucrează şi deplasarea rezultatului operaţiilor morfologice este dat de proprietăţile de invarianţă la translaţie:

$$A_t \oplus B = (A \oplus B)_t \text{ şi } A_t \ominus B = (A \ominus B)_\mathbf{t} \tag{6.11}$$

$$A \oplus B_\mathbf{t} = (A \oplus B)_{-\mathbf{t}} \text{ şi } A \ominus B_\mathbf{t} = (A \ominus B)_{-\mathbf{t}} \tag{6.12}$$

Aceste proprietăţi de invarianţă la translaţie (translaţia obiectului şi translaţia elementului structurant) pot avea mai multe interpretări. În primul rând, invarianţa la translaţie asigură faptul că forma rezultată prin dilatarea sau erodarea unei mulţimi este aceeaşi, indiferent de poziţia mulţimii prelucrate. În al doilea rând, relaţiile enunţate asigură suficienţa considerării doar a elementelor structurante ce conţin originea; un element structurant ce nu conţine originea poate fi obţinut prin translaţie dintr-un element structurant ce conţine originea; mulţimea (forma) rezultată în urma prelucrării va trebui translatată în sens opus elementului structurant.

În acelaşi grup de proprietăţi de invarianţă a operaţiilor morfologice se încadrează şi invarianţa la scalare (omotetie). Dacă λ este parametrul nenul de scalare, atunci:

$$\frac{1}{\lambda}(\lambda A \oplus B) = A \oplus \frac{1}{\lambda}B \text{ şi } \frac{1}{\lambda}(\lambda A \ominus B) = A \ominus \frac{1}{\lambda}B \tag{6.13}$$

Într-adevăr,

$$\frac{1}{\lambda}(\lambda A \oplus B) = \frac{1}{\lambda}(\lambda A + B^S) = \frac{1}{\lambda}\{\mathbf{x}|\mathbf{x} = \lambda\mathbf{a} - \mathbf{b}, \mathbf{a} \in A, \mathbf{b} \in B\} = \{\mathbf{x}|\mathbf{x} = \mathbf{a} - \frac{1}{\lambda}\mathbf{b}, \mathbf{a} \in A, \mathbf{b} \in B\} =$$

$$= A + \frac{1}{\lambda}B^S = A \oplus \frac{1}{\lambda}B$$

$$\frac{1}{\lambda}(\lambda A \ominus B) = \frac{1}{\lambda}(\lambda A^C \oplus B)^C = (\frac{1}{\lambda}(\lambda A^C \oplus B))^C = (A^C \oplus \frac{1}{\lambda}B)^C = A \ominus \frac{1}{\lambda}B$$

Relaţiile de invarianţă la scalare afirmă că rezultatul unei transformări morfologice aplicate unei versiuni scalate a formei A este identic cu aceeaşi scalare aplicată rezultatului transformării morfologice a formei A prin acelaşi element structurant scalat invers.

Proprietăţile de monotonie a unei transformări nu pot fi neglijate la descrierea acesteia. Atât dilatarea cât şi erodarea sunt crescătoare faţă de mulţimea ce se prelucrează (forma): dacă $A_1 \subseteq A_2$, atunci

$$A_1 \oplus B \subseteq A_2 \oplus B \text{ şi } A_1 \ominus B \subseteq A_2 \ominus B \tag{6.14}$$

Monotonia faţă de elementul structurant folosit este diferită: dilatarea este crescătoare, iar erodarea este descrescătoare: dacă $B_1 \subseteq B_2$, atunci:

$$A \oplus B_1 \subseteq A \oplus B_2 \text{ şi } A \ominus B2 \subseteq A \ominus B1 \tag{6.15}$$

Aceste proprietăţi subliniază corectitudinea interpretării dilatării ca o adăugare, ca o îngroşare, iar a erodării ca o subţiere; grosimea stratului adăugat sau îndepărtat este

dată de elementul structurant. Deci cu cât elementul structurant este mai mare, cu atât corpul se va mări mai mult în urma dilatării sau se va micşora mai mult în urma erodării.

În general, dilatarea este extensivă, adică:

$$A \subseteq A \oplus B. \tag{6.16}$$

O condiţie suficientă pentru ca această proprietate să fie adevărată este ca elementul structurant să conţină originea, $\{0_n\} \in B$. Atunci

$$A \oplus B = \bigcup_{\mathbf{b} \in B^S} A_{\mathbf{b}} = A \cup (\bigcup_{\mathbf{b} \in B^S - \{0_n\}} A_{\mathbf{b}}) \supseteq A \tag{6.17}$$

Condiţia ca elementul structurant să conţină originea nu este însă şi o condiţie necesară (a se vedea figura 6.7).

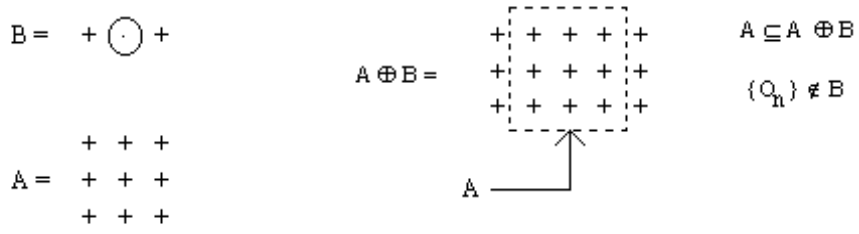

Fig. 6.7: Deşi elementul structurant nu îşi conţine originea, rezultatul dilatării include forma originală.

În general erodarea este antiextensivă, adică:

$$A \ominus B \subseteq A \tag{6.18}$$

O condiţie suficientă pentru ca această proprietate să fie adevărată este ca elementul structurant să conţină originea, $\{0_n\} \in B$. Atunci

$$A \ominus B = \bigcap_{\mathbf{b} \in B^S} A_{\mathbf{b}} = A \cap (\bigcap_{\mathbf{b} \in B^S - \{0_n\}} A_{\mathbf{b}}) \subseteq A \tag{6.19}$$

Condiţia ca elementul structurant să conţină originea nu este însă şi o condiţie necesară (a se vedea figura 6.8).

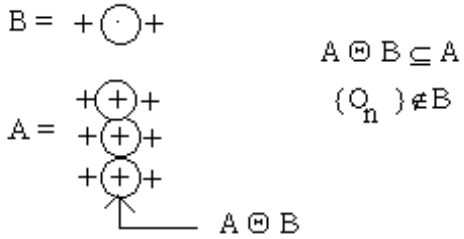

Fig. 6.8: Deşi elementul structurant nu îşi conţine originea, rezultatul erodării este inclus în mulţimea iniţială.

Dilatarea este pseudocomutativă, proprietate care se exprimă ca

$$A \oplus B = (B \oplus A)^S \tag{6.20}$$

$$A \oplus B = A + B^S = (A^S + B)^S = (B + A^S) = (B \oplus A)^S \tag{6.21}$$

Proprietatea de asociativitate a dilatării se scrie ca

$$A \oplus (B \oplus C) = (A \oplus B) \oplus C^S \tag{6.22}$$

Într-adevăr:

$$A \oplus (B \oplus C) = A + (B \oplus C)^S = A + (B + C^S)^S = A + B^S + C = (A \oplus B) + C = (A \oplus B) \oplus C^S \tag{6.23}$$

Erodarea nu este nici comutativă (sau pseudocomutativă) şi nici asociativă. Mai mult, "asociativitatea" erodării se scrie ca:

$$(A \ominus B) \ominus C = A \ominus (B \oplus C) \tag{6.24}$$

Într-adevăr:

$$(A \ominus B) \ominus C = ((A \ominus B)^C \oplus C)^C = ((A^C \oplus B) \oplus C)^C = (A^C \oplus (B \oplus C))^C = A \ominus (B \oplus C) \tag{6.25}$$

Proprietăţile de asociativitate ale dilatării şi erodării pot fi interpretate şi ca o "descompunere" a elementului structurant; dacă un element structurant poate fi considerat ca descompus prin dilatare în $X = B \oplus C$, atunci operaţia morfologică efectuată prin elementul structurant respectiv este echivalentă cu operaţiile morfologice prin "părţile" descompunerii elementului structurant, efectuate succesiv, adică:

$$A \oplus X = (A \oplus B) \oplus C^S \text{ şi } A \ominus X = (A \ominus B) \ominus C$$

Dilatarea şi erodarea sunt transformări (operaţii) ce nu păstrează numărul de conexităţi (numărul de obiecte). Rezultatul unei erodări poate fi o mulţime vidă (dacă elementul structurant nu poate fi inclus în formă pentru nici o poziţie) sau mai multe forme (componente conexe). Rezultatul dilatării unei mulţimi formate din mai multe componente conexe poate fi o singură componentă conexă; "găurile" conţinute într-o formă pot fi umplute.

Aceste proprietăţi ale erodării şi dilatării sunt fundamentul folosirii practice a acestora: prin erodare se pot elimina dintr-o mulţime componentele conexe ce sunt mai mici decât elementul structurant folosit (sau sunt mult diferite de acesta din punctul de vedere al formei) - aplicaţiile ce ar putea folosi o asemenea comportare sunt separarea obiectelor după formă şi dimensiune şi eliminarea zgomotului suprapus scenei. Prin dilatare se pot grupa într-o singură entitate obiecte apropiate şi se pot umple golurile înglobate în obiecte.

Dilatarea şi erodarea nu sunt operaţii inversabile (nu admit o transformare inversă). Cu atât mai mult, dilatarea nu este inversa erodării şi erodarea nu este inversa dilatării; exemplele prezentate în figurile 6.9 şi 6.10 ilustrează această afirmaţie.

$$(A \ominus B) \oplus B \neq A \text{ şi } (A \oplus B) \ominus B \neq A \tag{6.26}$$

O altă categorie de proprietăţi se referă la distributivitatea operaţiilor morfologice faţă de operaţiile cu mulţimi. Pentru dilatare avem:

$$(A \cup B) \oplus C = (A \oplus C) \cup (B \oplus C) \tag{6.27}$$

$$(A \oplus C) \cup (B \oplus C) = (\bigcup_{\mathbf{c} \in C^S} A_{\mathbf{c}}) \cup (\bigcup_{\mathbf{c} \in C^S} B_{\mathbf{c}}) = (\bigcup_{\mathbf{c} \in C^S} A_{\mathbf{c}} \cup B_{\mathbf{c}}) = \bigcup_{\mathbf{c} \in C^S} (A \cup B)_{\mathbf{c}} = (A \cup B) \oplus C$$

$$A \oplus (B \cup C) = (A \oplus B) \cup (A \oplus C) \tag{6.28}$$

$$(A \oplus B) \cup (A \oplus C) = (A + B^S) \cup (A + C^S) = (B^S + A) \cup (C^S + A) = (\bigcup_{\mathbf{a} \in A} B^S_{\mathbf{a}}) \cup (\bigcup_{\mathbf{a} \in A} C^S_{\mathbf{a}}) =$$

$$= \bigcup_{\mathbf{a} \in A} (B^S_{\mathbf{a}} \cup C^S_{\mathbf{a}}) = \bigcup_{\mathbf{a} \in A} (B \cup C)^S_{\mathbf{a}} = (B \cup C)^S + A = A + (B \cup C)^S = A \oplus (B \cup C)$$

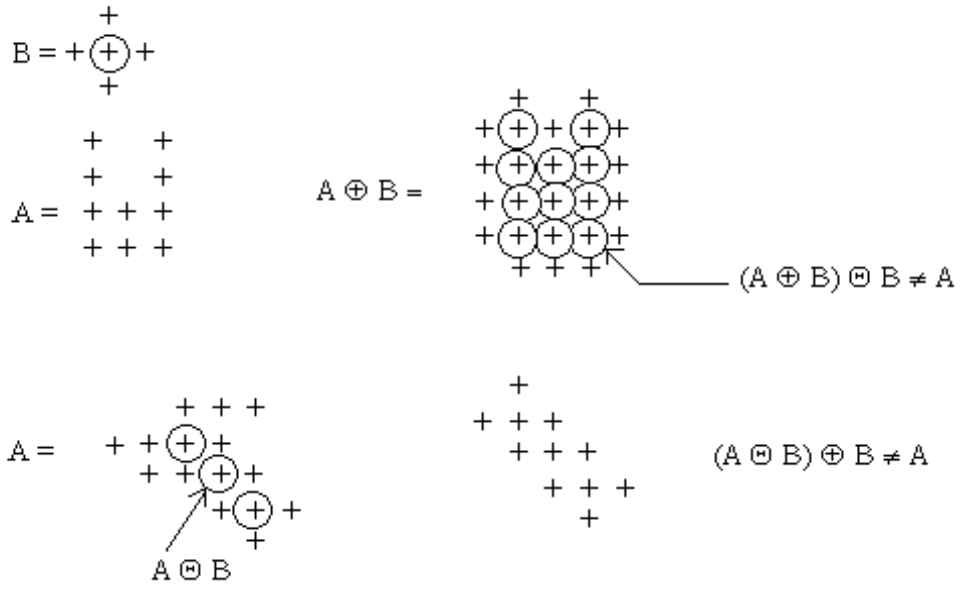

Fig. 6.9: Dilatarea şi erodarea nu sunt transformări inverse una alteia; ilustrare pentru mulţimi discrete.

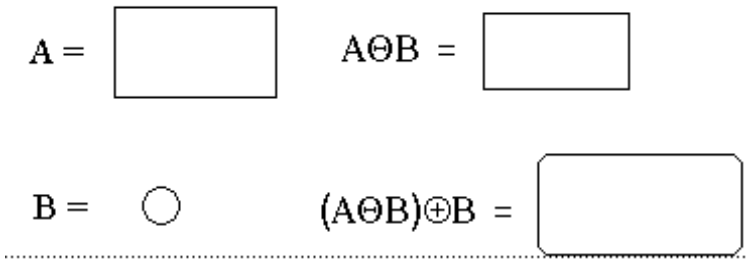

Fig. 6.10: Dilatarea şi erodarea nu sunt transformări inverse una alteia; ilustrare pentru mulţimi continue.

Această din urmă relaţie poate fi interpretată şi ca o descompunere prin reuniune a elementului structurant $X = B \cup C$, caz în care operaţia morfologică efectuată prin elementul structurant respectiv este echivalentă cu operaţiile morfologice prin "părţile" descompunerii elementului structurant, efectuate succesiv, adică:

$$A \oplus X = (A \oplus B) \cup (A \oplus C) \tag{6.29}$$

Erodarea are proprietăţi asemănătoare dilatării faţă de intersecţia mulţimilor:

$$A \ominus (B \cup C) = (A \ominus B) \cap (A \ominus C) \tag{6.30}$$

$$(A \ominus B) \cap (A \ominus C) = (\bigcap_{\mathbf{t} \in B^S} A_{\mathbf{t}}) \cap (\bigcap_{\mathbf{t} \in C^S} A_{\mathbf{t}}) = \bigcap_{\mathbf{t} \in B^S \cup C^S} A_{\mathbf{t}} = \bigcap_{\mathbf{t} \in (B \cup C)^S} A_{\mathbf{t}} = A \ominus (B \cup C)$$

$$(A \cap B) \ominus C = (A \ominus C) \cap (B \ominus C)$$

$$(A \ominus C) \cap (B \ominus C) = (\bigcap_{\mathbf{c} \in C^S} A_{\mathbf{c}}) \cap (\bigcap_{\mathbf{c} \in C^S} B_{\mathbf{c}}) = \bigcap_{\mathbf{c} \in C^S} (A_{\mathbf{c}} \cap B_{\mathbf{c}}) = \bigcap_{\mathbf{c} \in C^S} (A \cap B)_{\mathbf{c}} = (A \cap B) \ominus C$$

Înrudite cu proprietăţile de distributivitate deja enunţate, există o serie de inegalităţi (relaţii de incluziune) de aceeaşi natură:

$$A \oplus (B \cap C) \subseteq (A \oplus B) \cap (A \oplus C) \text{ şi } (B \cap C) \oplus A \subseteq (B \oplus A) \cap (C \oplus A)$$

$$A \ominus (B \cap C) \supseteq (A \ominus B) \cup (A \ominus C) \text{ şi } (B \cap C) \ominus A \supseteq (B \ominus A) \cap (C \ominus A)$$

Această tratare [aproape] exhaustivă a proprietăţilor operaţiilor morfologice de bază (dilatare, erodare) se regăseşte în majoritatea monografiilor de morfologie matematică, ca de exemplu în [14], [7]. Deşi, în esenţă, proprietăţile tratate sunt aceleaşi, se impun câteva precizări.

Notaţiile introduse în acest text sunt diferite pentru operaţiile Minkowski (adunare $+$ şi scădere $-$) şi pentru operaţiile morfologice (dilatare $\oplus$ şi erodare $\ominus$). Legătura dintre aceste operaţii este dată de:

$$A \oplus B = A + B^S \text{ şi } A \ominus B = A - B^S \tag{6.31}$$

În majoritatea lucrărilor de morfologie însă, nu se face nici o distincţie de notaţie între aceste operaţii, ceea ce poate genera confuzii. Chiar în condiţiile păstrării aceloraşi notaţii, există abordări diferite privind definirea operaţiilor morfologice. În [7] dilatarea şi erodarea sunt identice cu adunarea, respectiv scăderea Minkowski (deci fără simetrizarea elementului structurant). Avantajul unei asemenea formalizări este existenţa proprietăţilor de comutativitate şi asociativitate a dilatării (determinate direct de comutativitatea şi asociativitatea adunării Minkowski). În plus, rezultatul dilatării este o "creştere" a formei pe direcţia şi în sensul elementului structurant folosit.

Dezavantajul acestei abordări este acela că trebuie redefinit conceptul de dualitate 6.8 (dacă se doreşte păstrarea proprietăţii fundamentale de dualitate a dilatării şi erodării), după cum urmează:

$$Dual(Op(A,B))|_{B\ parametru} = (Op(A^C, B^S))^C \tag{6.32}$$

În [14] este preferată definirea dilatării unei mulţimi ca adunarea Minkowski a mulţimii cu elementul structurant simetrizat.

6.1.5 Aspecte de implementare

Din punct de vedere practic (al implementării), elementul structurant poate fi interpretat analog suportului ferestrei de filtrare folosită pentru orice operaţie bazată pe principiul ferestrei glisante: cu valorile selectate din imagine "se face ceva". Va trebui însă introdus un mecanism de specificare a formei ferestrei de filtrare (elementului structurant) care să permită modificarea relativ simplă a acesteia (puterea morfologiei matematice constă în alegerea a diferite tipuri de elemente structurante optime unei anumite prelucrări). Convenţia de reprezentare a imaginilor binare este de a avea asociate valori nule punctelor de fundal şi valori pozitive (1 sau 255) punctelor obiect[1]. În aceste condiţii, putem găsi o echivalenţă intuitivă simplă operaţiilor morfologice de erodare şi dilatare. Rezultatul operaţiei de erodare într-un pixel este nenul dacă şi numai dacă elementul structurant plasat cu originea în acel punct este inclus în forma de prelucrat, şi deci, dacă şi numai dacă toate valorile selectate de elementul structurant sunt nenule; aceasta înseamnă că putem implementa operaţia de erodare printr-o operaţie de minim. Rezultatul operaţiei de dilatare într-un pixel este nul dacă şi numai dacă elementul structurant plasat cu originea în acel punct nu are nici un punct comun cu forma de prelucrat, şi deci, dacă şi numai dacă toate valorile selectate de elementul structurant sunt nule; aceasta înseamnă că putem implementa operaţia de dilatare printr-o operaţie de maxim. În cele ce urmează prezentăm codul pentru operaţia de erodare.

```
# define SIZE_MAX_ES=16;
```

[1]Reprezentarea grafică a acestei imagini, realizată cu tabela de nivele de gri normală va afişa deci obiecte albe pe un fundal negru.

```
int min,min_i,max_i,min_j,max_j;
integer es[SIZE_MAX_ES][2];
min_i=0;max_i=0,min_j=0,max_j=0;
for (k=0;k<dim_es;k++) {
  if (min_i > es[k][1])
    min_i=es[k][1];
  if (max_i < es[k][1])
    max_i=es[k][1];
  if (min_j > es[k][2])
    min_j=es[k][2];
  if (max_j < es[k][2])
    max_j=es[k][2]; }
for (i=-min_i;i<NRLIN-max_i;i++)
 for (j=-min_j;j<NRCOL-max_j;j++) {
   min=MAXINT;
   for (k=0;k<dim_es;k++)
     if (min > img[i+es[k][1]][j+es[k][2]]) then
       min=img[i+es[k][1]][j+es[k][2]];
   img_out[i][j]=min;  }
```

Elementul structurant folosit este reprezentat prin matricea de întregi *es*, având cel mult 16 linii şi 2 coloane. Fiecare linie a matricii corespunde unui punct din elementul structurant, reprezentat prin coordonatele faţă de originea elementului structurant: *es[k][1]* şi *es[k][2]*. De exemplu, elementul structurant orizontal, centrat, având trei puncte este reprezentat prin $es = [0\ -1; 0\ 0; 0\ 1]^T$. Numărul de puncte ce compun elementul structurant este *dim_ es*. Prima buclă *for* determină dimensiunile dreptunghiului de încadrare a elementului structurant, pentru a putea evita efectele de margine la prelucrarea imaginii.

6.2 Transformări morfologice derivate

Vom numi operaţie (transformare) morfologică derivată operaţia morfologică construită ca o combinaţie de transformări de bază: erodări, dilatări şi operaţii clasice ansambliste.

6.2.1 Deschiderea şi închiderea

După cum s-a arătat, erodarea şi dilatarea nu sunt transformări inverse una alteia (deci alternarea lor va produce un rezultat diferit de mulţimea originală ce a fost prelucrată). Această observaţie conduce la ideea utilizării unor iterări ale operaţiilor morfologice de

bază, obţinând astfel operaţii mai complexe, ale căror proprietăţi le fac mai adecvate utilizării în scopuri practice.

Deschiderea morfologică a mulţimii A prin elementul structurant B se defineşte ca erodarea mulţimii cu elementul structurant respectiv, urmată de dilatarea cu elementul structurant simetrizat:

$$A \bigcirc B = (A \ominus B) \oplus B^S \tag{6.33}$$

Închiderea morfologică a mulţimii A prin elementul structurant B se defineşte ca dilatarea mulţimii cu elementul structurant respectiv, urmată de erodarea cu elementul structurant simetrizat:

$$A \bullet B = (A \oplus B) \ominus B^S \tag{6.34}$$

Proprietatea de bază a deschiderii şi închiderii morfologice este aceea că sunt transformări duale una alteia (proprietate ce derivă din dualitatea blocurilor constituente de bază, dilatarea şi erodarea). Această proprietate permite interpretarea rezultatelor unei operaţii şi deducerea proprietăţilor acesteia pe baza caracteristicilor dualei sale.

$$(A \bigcirc B)^C = A^C \bullet B \text{ şi } (A \bullet B)^C = A^C \bigcirc B \tag{6.35}$$

Demonstraţia acestor proprietăţi este imediată:

$$(A^C \bigcirc B)^C = ((A^C \ominus B) \oplus B^S)^C = ((A \oplus B)^C \oplus B^S)^C = (A \oplus B) \ominus B^S = A \bullet B$$

$$(A^C \bullet B)^C = ((A^C \oplus B) \ominus B^S)^C = ((A \ominus B)^C \ominus B^S)^C = (A \ominus B) \oplus B^S = A \bigcirc B$$

În mod evident, rezultatul unei deschideri sau al unei închideri este diferit de mulţimea ce a fost prelucrată. Relaţia dintre rezultatul prelucrării şi mulţimea iniţială este dată de proprietăţile de extensivitate ale transformărilor.

Deschiderea morfologică este antiextensivă, adică $A \bigcirc B \subseteq A$

Închiderea morfologică este extensivă, adică $A \subseteq A \bullet B$

Deci, pentru a sintetiza, putem afirma că:

$$A \bigcirc B \subseteq A \subseteq A \bullet B \tag{6.36}$$

Relaţiile pot fi interpretate ca o modificare sigură a mulţimii: prin deschidere se vor elimina o parte dintre elementele mulţimii ce se prelucrează, iar prin închidere se adaugă elemente noi mulţimii.

Proprietatea de idempotenţă a operaţiilor introduce o limitare a modificărilor: mulţimea obţinută după o deschidere sau închidere este invariantă la repetarea operaţiei:

$$(A \bullet B) \bullet B = A \bullet B \text{ şi } (A \bigcirc B) \bigcirc B = A \bigcirc B \tag{6.37}$$

Relaţiile se demonstrează folosind proprietăţile deja enunţate ale deschiderii şi închiderii (extensivitate) şi proprietăţile de monotonie ale operaţiilor morfologice de bază.

Închiderea şi deschiderea moştenesc o parte dintre proprietăţile operaţiilor morfologice de bază: invarianţa la translaţie şi la scalare, monotonia faţă de mulţimea prelucrată şi faţă de elementul structurant folosit. Din punctul de vedere al acestor proprietăţi, deschiderea se comportă analog erodării iar închiderea are o comportare analoagă dilatării.

$$A_{\mathbf{t}} \bigcirc B = (A \bigcirc B)_{\mathbf{t}} \text{ şi } A_{\mathbf{t}} \bullet B = (A \bullet B)_{\mathbf{t}}$$

$$\tfrac{1}{\lambda}(\lambda A \bigcirc B) = A \bigcirc \tfrac{1}{\lambda} B \text{ şi } \tfrac{1}{\lambda}(\lambda A \bullet B) = A \bullet \tfrac{1}{\lambda} B$$

$$\text{Dacă } A_1 \subseteq A_2 : A_1 \bigcirc B \subseteq A_2 \bigcirc B \text{ şi } A_1 \bullet B \subseteq A_2 \bullet B.$$

$$\text{Dacă } B_1 \subseteq B_2 : A \bigcirc B_2 \subseteq A \bigcirc B_1 \text{ şi } A \bullet B_1 \subseteq A \bullet B_2$$

În ceea ce priveşte proprietăţile legate de comportarea faţă de translaţie a operatorilor de deschidere şi închidere, merită subliniat faptul că proprietatea este identică cu cea a erodării şi dilatării doar la translaţia mulţimii prelucrate (rezultatul unei deschideri sau închideri a unei mulţimi este acelaşi, indiferent de poziţia spaţială a mulţimii). În cazul translatării elementului structurant, rezultatul operaţiei este acelaşi, invariant la translaţie (ca rezultat a iterării erodării şi dilatării cu elemente structurante simetrice).

Pentru realizarea efectivă a operaţiilor de deschidere şi închidere este importantă exprimarea acestora ca operaţii la nivelul elementelor mulţimilor ce se prelucrează (şi nu ca o iterare de operaţii morfologice de bază). Deschiderea mai poate fi exprimată şi ca mulţimea elementelor structurante translatate ce sunt incluse în mulţimea de prelucrat:

$$A \bigcirc B = \bigcup_{B_{\mathbf{x}} \subseteq A} B_{\mathbf{x}} \tag{6.38}$$

Închiderea mai poate fi exprimată şi ca mulţimea punctelor pentru care toate elementele structurante translatate ce le conţin au puncte comune cu mulţimea de prelucrat:

$$A \bullet B = \bigcap_{B_{\mathbf{x}} \cap A \neq \emptyset} B_{\mathbf{x}} \tag{6.39}$$

Pe baza acestor exprimări se pot deduce şi efectele practice ale deschiderii şi închiderii asupra formelor (mulţimilor). Prin deschidere, formele mai mici ca elementul structurant folosit vor fi eliminate, se lărgesc golurile înglobate în obiecte, contururile sunt netezite prin teşirea convexităţilor iar obiectele unite prin istmuri sunt separate. Datorită proprietăţii de dualitate, închiderea va avea aceleaşi efecte asupra fundalului (complementarei obiectelor) pe care le are deschiderea asupra mulţimilor. Închiderea umple golurile înglobate în obiecte (dacă aceste găuri sunt mai mici decât elementul structurant folosit), netezeşte contururile formelor prin umplerea concavităţilor şi uneşte obiectele foarte apropiate (umple "strâmtorile").

Efectele operaţiilor de deschidere şi închidere pot fi considerate ca analoage efectelor unei filtrări de netezire a formelor şi eliminare a zgomotului (zgomot interpretat ca obiecte şi găuri de mici dimensiuni). În [14], în cadrul teoriei algebrice a morfologiei matematice, un filtru este definit ca o operaţie crescătoare şi idempotentă. Trebuie făcută deci distincţia între filtrul algebric şi filtrul obişnuit, în sensul teoriei prelucrării semnalelor.

6.2.2 Filtrele alternate secvenţial

Filtrele morfologice sunt transformari neliniare ale semnalului, care modifică local caracteristici geometrice (de formă). Teoria algebrică generală a filtrelor [14] defineşte o transformare ca filtru dacă aceasta este crescătoare şi idempotentă, ceea ce înseamnă deci că deschiderea şi închiderea sunt filtre, în timp ce erodarea şi dilatarea nu sunt filtre.

Dacă o imagine (mulţime) este filtrată prin deschideri după elemente structurante (discuri de rază r) crescătoare, aceasta este echivalent cu a aplica deschiderea după elementul structurant cu raza cea mai mare. Totuşi, se pot construi filtre ce actionează asupra obiectelor şi detaliilor în mod gradat, pe masura iterării: aceste sunt filtrele alternate secvenţial, ce alternează deschideri şi închideri după elemente structurante de dimensiune crescătoare:

$$FAS(A) = (((((A \bigcirc B) \bullet B) \bigcirc 2B) \bullet 2B) \bigcirc 3B) \bullet ... \tag{6.40}$$

sau

$$FAS(A) = (((((A \bullet B) \bigcirc B) \bullet 2B) \bigcirc 2B) \bullet 3B) \bigcirc ... \tag{6.41}$$

6.2.3 Operatori morfologici de contur

O problema curentă a prelucrărilor de imagini este extragerea punctelor de contur. Cazul în care imaginea este binară constituie o simplificare a metodelor şi calculelor. Păstrând

interpretarea pixelilor ca fiind puncte ale obiectului sau ale fundalului, punctele de contur sunt acele puncte de obiect ce au cel puţin un punct de fundal vecin. În general, vecinătatea folosită este cea de tip disc unitar, a cărei formă va depinde puternic de tipul de metrică folosită.

Există trei extractoare morfologice de contur: conturul interior (6.42), conturul exterior (6.43) şi gradientul morfologic (6.44).

$$\delta A = A - A \ominus B \tag{6.42}$$

$$\Delta A = A \oplus B - A \tag{6.43}$$

$$grad\ A = A \oplus B - A \ominus B \tag{6.44}$$

6.3 Extinderea morfologiei matematice la nivele de gri

Pâna în acest moment s-au prezentat operaţiile morfologice clasice, adică aplicate asupra unor seturi (mulţimi). Acestea prezintă limitarea implicită în aplicarea numai asupra unei categorii particulare de imagini, şi anume cele a căror structură poate fi uşor şi imediat asociată unor mulţimi, adică imaginile binare. Pentru extinderea operaţiilor morfologice la funcţii se vor construi transformări ce permit trecerea de la o reprezentare de tip funcţie la o reprezentare de tip mulţime.

În cele ce urmează ne vom referi la o submulţime A a spaţiului discret n-dimensional, adică $A \subseteq \mathbf{Z}^n$. Elementele mulţimii A sunt puncte, reprezentate prin vectorul n-dimensional al coordonatelor, $\mathbf{x} \in A$, cu $\mathbf{x} = (x_1, x_2, ..., x_n)$. Ordinea coordonatelor este arbitrară.

Vom numi primele $n-1$ coordonate ale unui punct $\mathbf{x}$ *domeniu spaţial* al mulţimii şi cea de-a n-a coordonată, *suprafaţa mulţimii*. Vom nota aceste "părţi" ale mulţimii A cu $A_{(1,n-1)}$ şi respectiv $A_{(n,n)}$. Putem avea n moduri diferite de a alege aceste partiţii ale coordonatelor (corespunzând modului în care se poate alege o coordonată dintr-un total de n).

Vârful (top) sau s*uprafaţa de vârf* a unei mulţimi A este funcţia $T(A) : A_{(1,n-1)} \longrightarrow A_{(n,n)}$ definită de:

$$T(A)(\mathbf{z}) = \max\{y \mid (\mathbf{z}, y) \in A, \forall \mathbf{z} \in A_{(1,n-1)}\} \tag{6.45}$$

În acest mod se introduce o funcţie, pornind de la o mulţime; funcţia este cu valori întregi (funcţie scalară) de argument vectorial $n-1$-dimensional. Pentru exemplificare considerăm cazul în care $n = 2$, deci fiecare punct al mulţimii A este un vector bidimensional, $\mathbf{x} = (i, j)$. Considerăm coordonata i ca fiind domeniul spaţial şi coordonata j ca

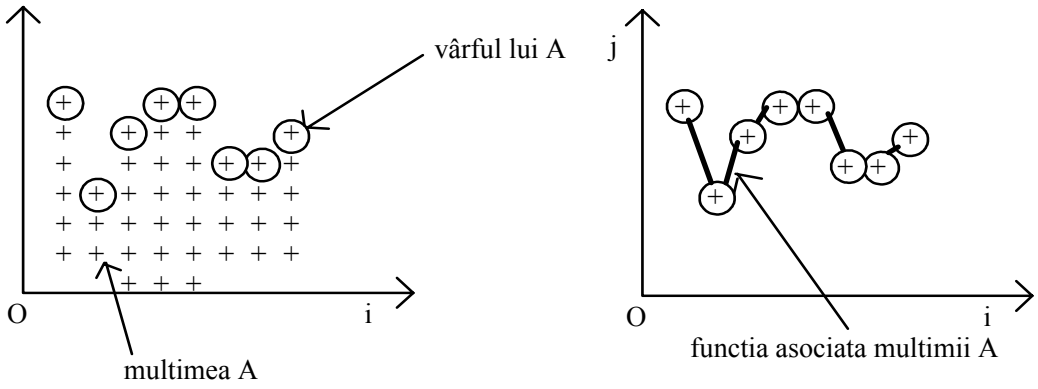

Fig. 6.11: Exemplificare pentru funcţia vârf a unei mulţimi.

fiind suprafaţa mulţimii. Atunci, prin transformarea vârf, asociem mulţimii A o funcţie definită pe domeniul său spaţial şi cu valori în suprafaţa sa.

Umbra unei funcţii oarecare f este definită ca transformarea $f : B \subseteq \mathbf{Z}^{n-1} \longrightarrow C \subseteq \mathbf{Z}$ dată de:

$$U(f) = \{(\mathbf{z}, y) \mid \mathbf{z} \in B, y \in \mathbf{Z},\ y \leqslant f(\mathbf{z})\} \tag{6.46}$$

În acest mod, plecând de la o funcţie de variabilă vectorială $n-1$-dimensională, cu valori scalare, se obţine o mulţime ale cărei elemente sunt vectori n-dimensionali. Mulţimea obţinută este semideschisă (nemarginită).

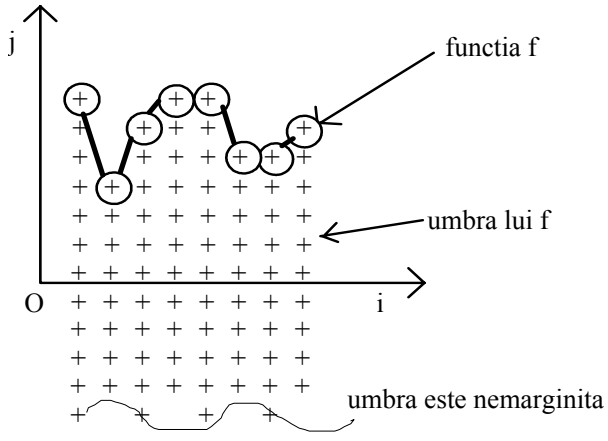

Fig. 6.12: Determinarea umbrei unei mulţimi.

În mod evident, vom avea următoarele relaţii între cele două transformări mulţime-funcţie (vârful şi umbra):

$$T(U(F)) = f \text{ şi } A \subseteq U(T(A)) \tag{6.47}$$

Operaţiile morfologice pentru funcţii se definesc prin intermediul transformării acestora în mulţimi (reducând astfel operaţiile pe funcţii la operaţii pe mulţimi, aşa cum au fost prezentate la morfologia binară):

$$f \ominus g = T\left(U(f) \ominus U(g)\right) \text{ şi } f \oplus g = T\left(U(f) \oplus U(g)\right) \tag{6.48}$$

Ca şi în cazul morfologiei pe mulţimi, g se numeşte element structurant (funcţie structurantă). Se numeşte element structurant *flat*, elementul structurant pentru care $g(\mathbf{y}) = 0, \forall \mathbf{y} \in Supp(g)$. Aceste elemente structurante de tip *flat* sunt echivalente cu cele folosite în morfologia clasică pe mulţimi.

Forma echivalentă a definiţiilor erodării şi dilatării (folosite în mod efectiv în practică) este, pentru $\forall \mathbf{x} \in Supp(f)$:

$$f \ominus g = \min_{\mathbf{y} \in Supp(g)} \left\{ f(\mathbf{x} - \mathbf{y}) - g(\mathbf{y}) \right\}, \; f \oplus g = \max_{\mathbf{y} \in Supp(g)} \left\{ f(\mathbf{x} - \mathbf{y}) + g(\mathbf{y}) \right\} \tag{6.49}$$

Dacă se utilizează elemente structurante plate (deci caracterizate doar de forma suportului, şi nu şi de valori asociate acestora), cele două relaţii de definiţie devin identice cu filtrările de ordine de rang minim şi respectiv maxim:

$$f \ominus g = \min_{\mathbf{y} \in Supp(g)} \left\{ f(\mathbf{x} - \mathbf{y}) \right\}, \; f \oplus g = \max_{\mathbf{y} \in Supp(g)} \left\{ f(\mathbf{x} - \mathbf{y}) \right\} \tag{6.50}$$

Capitolul 7

METODE DE COMPRESIE A IMAGINILOR

Termenul de compresie a imaginilor (uneori numit şi codare a imaginilor) se referă la o clasă largă de tehnici şi metode al căror scop este reprezentarea unui imagini date cu un număr cât mai mic de biţi (mai mic decât numărul de biţi al reprezentării iniţiale). Necesitatea reducerii cantităţii de informaţie necesară reprezentării este evidentă dacă considerăm cazul memorării imaginilor radiografice (4000 x 2500 pixeli, cu 4096 nivele de gri, deci 14,3 MB) sau al transmisiei de televizune alb-negru (625 x 625 pixeli cu 256 nivele de gri, de 50 de ori pe secundă, deci un flux de 18.6 MB/ secundă) [9]. Procesul de recompunere a imaginii iniţiale din reprezentarea restrânsă se numeşte decompresie sau decodare; este evident că prin decodare trebuie să se obţină o imagine cât mai apropiată de imaginea orginală. Există două categorii fundamentale de tehnici de compresie (codare): codarea fără pierderi (în care imaginea decodată este identică cu imaginea iniţială) şi codarea cu pierderi, în care se admit mici diferenţe faţă de original. Calitatea unui procedeu de compresie (pentru o imagine dată) se măsoară prin factorul de calitate (raportul semnal zgomot (3.4) dintre imaginea originală f şi imaginea decodată g) şi factorul (raportul) de compresie. Factorul de compresie C este raportul dintre cantitatea de informaţie necesară reprezentării imaginii iniţiale şi cantitatea de informaţie necesară reprezentării imaginii codate; evident compresia are loc dacă factorul de compresie este supraunitar ($C > 1$). Uneori, factorului de compresie i se asociază (sau este înlocuit de) rata de compresie: cantitatea de informaţie necesară reprezentării comprimate a fiecărui pixel al imaginii; rata de compresie se măsoară în biţi per pixel (bpp).

O altă clasificare posibilă a tehnicilor de compresie se poate face după tipul imaginii căreia i se aplică: vom face astfel distincţia între compresia imaginilor binare şi compresia imaginilor cu nivele de gri. Se impune totuşi o observaţie: metodele de codare ce se vor prezenta în cadrul tehnicilor speifice imaginilor binare pot fi folosite pentru compresia oricărei succesiuni de valori binare, indiferent de semnificaţia acestora (ceea ce înseamnă că ar putea fi folosite şi pentru compresia imaginilor cu nivele de gri) şi sunt metode de

compresie fără pierderi.

7.1 Compresia imaginilor binare

Putem considera că singura categorie de imagini binare de interes sunt imaginile în alb-negru (sau monocrome); valorile punctelor acestora sunt fie 0 (reprezentând fundalul de culoare albă), fie 1 (reprezentând punctele de interes, de culoare neagră)[1]. Cele două clase de metode de codare pe care le avem în vedere sunt codarea entropică (metoda de codare Huffman) şi metodele de codare *on-line* (pe flux de biţi); deosebirea dintre aceste metode (la un nivel al implementării) este că pentru codarea entropică este necesară parcurgerea şi stocarea intermediară a întregii imagini.

7.1.1 Codarea entropică (Huffman)

Codarea entropică (Huffman) este metoda optimală de codare a unei surse de informaţie. Codarea sursei de informaţie S ale cărei mesaje sunt $\{s_1, s_2, ..., s_N\}$, de probabilităţi $\{p(s_1), p(s_2), ..., p(s_N)\}$ prin alfabetul X cu D simboluri înseamnă a asocia fiecărui mesaj s_i al sursei primare de informaţie un şir de simboluri din alfabetul codului. Lungimea medie a cuvintelor de cod este dată de raportul dintre entropia sursei primare şi entropia alfabetului codului:

$$\bar{l} = \frac{H(S)}{H(X)} \tag{7.1}$$

Se doreşte obţinerea unei lungimi medii minime a cuvintelor de cod, şi deci, echivalent, mărirea entropiei alfabetului codului; la limită, lungimea media minimă posibilă de obţinut este:

$$\bar{l}_{\min} = \frac{H(S)}{\log D} \tag{7.2}$$

Procedeul practic prin care se realizează alocarea simbolurilor din alfabetul codului astfel încât entropia acestuia să fie maximizată (metoda Huffman) se bazează pe reducerea iterativă a numărului de simboluri ale sursei de codat şi construirea unei surse restrânse. La fiecare etapă cele D simboluri cele mai puţin probabile ale sursei de informaţie sunt reunite într-un nou simbol; procedeul de restrângere continuă până când se obţine o sursă redusă cu exact D simboluri. Apoi, pentru fiecare reunire de simboluri, fiecare mesaj individual va primi codul cuvântului reunit ca prefix, urmat de câte un simbol din alfabetul codului. Vom considera următorul exemplu.

O sursă S generează 6 simboluri, de probabilităţi descrise de vectorul $P = [0.3; 0.25; 0.2; 0.1; 0.1; 0.05]$. *Sursa este codată optimal, simbol cu simbol, cu cuvinte de cod generate cu*

[1] Deci convenţia de reprezentare prin culori este modificată faţă de convenţia generală utilizată – 0 nu mai este negru, ci alb.

simboluri dintr-un alfabet ternar. Să se calculeze cuvintele de cod, arborele de codare şi eficienţa codării.

Codarea optimală a unei surse se realizează conform algoritmului Huffman. Se ştie că numărul de simboluri ale sursei ce se codează trebuie să îndeplinească o anume relaţie $\left(\frac{N-D}{D-1} \in \mathbf{N}\right)$; în acest caz, $N = 6$, $D = 3$ şi deci

$$\frac{N-D}{D-1} = \frac{6-3}{3-1} = \frac{3}{2} \notin \mathbf{N}$$

Completarea se face adăugând sursei simboluri de probabilitate nulă; în acest caz, cu un singur astfel de simbol adăugat obţinem $N = 7$ şi

$$\frac{N-D}{D-1} = \frac{7-3}{3-1} = \frac{4}{2} \in \mathbf{N}$$

Entropia sursei extinse S este dată de:

$$H(S) = -\sum_{i=1}^{7} p(s_i) \log p(s_i) = -\left(0.3 \log 0.3 + 0.25 \log 0.25 + 0.2 \log 0.2 + 2 \cdot 0.1 \log 0.1\right) -$$

$$-0.05 \log 0.05 - 0 \log 0 = 2.366 \text{ bit/simbol}$$

Atunci, conform (7.2), lungimea medie minimă a unui cuvânt de cod este:

$$\bar{l}_{\min} = \frac{H(S)}{\log D} = \frac{2.366}{\log 3} = 1.493 \text{ bit}$$

Codarea Huffman este realizată conform tabelului 7.1.

Sursă primară	$p(s_i)$	Cod	Sursă restrânsă	$p(r_{ji})$	Cod	Sursă restrânsă	$p(r_{ji})$	Cod
						r_{21}	0.45	0
s_1	0.3		r_{11}	0.3		r_{22}	0.3	1
s_2	0.25		r_{12}	0.25		r_{23}	0.25	2
s_3	0.2		r_{13}	0.2	00			
			r_{14}	0.15	01			
s_4	0.1		r_{15}	0.1	02			
s_5	0.1	010						
s_6	0.05	011						
s_7	0	012						

Tabel 7.1: Codare Huffman

Cuvintele de cod sunt grupate în tabelul 7.2; pe baza lor putem calcula lungimea medie a cuvintelor de cod:

$$\overline{l} = \sum_{i=1}^{7} l_i p(s_i) = 1 \cdot 0.3 + 1 \cdot 0.25 + 2 \cdot 0.2 + 2 \cdot 0.1 + 3 \cdot 0.1 + 3 \cdot 0.05 + 3 \cdot 0 = 1.6$$

Sursă primară	$p(s_i)$	Cod	Lungime
s_1	0.3	1	1
s_2	0.25	2	1
s_3	0.2	00	2
s_4	0.1	02	2
s_5	0.1	010	3
s_6	0.05	011	3
s_7	0	012	3

Tabel 7.2: Coduri asociate simbolurilor sursei şi lungimile lor

În cazul imaginilor (sau al şirurilor binare) mesajele sursei primare sunt formate din grupuri de câte B biţi succesivi (astfel, sursa primară are 2^B mesaje, ale căror probabilităţi de apariţie sunt specifice imaginii considerate). Cu cât B este mai mare, cu atât mai mare va fi eficienţa codării. Tabelul de codare (echivalenţa între mesajele sursei primare şi şirurile de simboluri de cod) este specific fiecărei imagini şi trebuie să însoţească codul. Codarea clasică este binară ($D = 2$, $X = \{0, 1\}$).

7.1.2 Codarea pe flux de biţi

Metodele de codare pe flux de biţi (*on-line*) se regăsesc la primul nivel de codare al transmisiilor de fax (RLE, WBS) sau ca tehnici generale de compresie a datelor, înglobate atât în formate de fişiere grafice (TIFF) cât şi în arhivatoare uzuale (ZIP) – metoda Ziv-Lempel. Aplicarea codării la imagini presupune în primul rând transformarea imaginii (structurii de matrice a acesteia) într-un vector (şir) prin concatenarea liniilor acesteia.

Codarea RLE

Principiul codării RLE (*Run Length Encoding*) este acela de a memora numărul de simboluri succesive ale şirului binar ce au aceeaşi valoare. Odată ce o secvenţă de simboluri succesive de aceeaşi valoare (*run length*) se încheie, simbolul următor nu poate avea decât

valoarea complementară celei iniţiale, prin alternanţă. Şirul codat trebuie să înceapă cu valoarea primului simbol.

Conform acestei codări, şirul de valori binare $0,1,1,1,1,0,0,0,1,0,1,1,1,0,1,1,1$ este codat RLE ca $0,4,3,1,1,3,1,3$, iar şirul binar $1,1,1,1,1,1,1,0,1,0,0,0,0,0,0$ este codat RLE ca $1,7,1,1,5$. Şirul cuvintelor de cod trebuie deci să fie compus din codurile ce corespund lungimilor secvenţelor de valori identice succesive; aceasta înseamnă că numărul de biţi al reprezentării binare al cuvintelor de cod trebuie să permită reprezentarea lungimii celei mai mari secvenţe. Cum determinarea de fiecare dată a acestei lungimi maximale şi varierea lungimii cuvântului de cod nu este o soluţie corespunzătoare, se preferă fixarea unei lungimi maximale admise a secvenţelor. Orice secvenţă mai lungă decât secvenţa maximă este despărţită prin intercalarea fictivă a unor simboluri complementare. Să considerăm o codare RLE cu lungimea cuvântului de cod de 2 biţi; aceasta înseamnă că lungimea secvenţei maxime este 3 (codul 0 trebui rezervat pentru secvenţa de lungime nulă). Dacă şirul binar este $1,1,1,1,1,1,1,0,1,0,0,0,0,0$ atunci şirul codat va fi 1, 3, 0, 3, 0, 1, 1, 1, 3, 0, 2 (unde fiecare cifră va fi reprezentată pe 2 biţi).

O codare mai bună se obţine dacă lungimile secvenţelor de simboluri identice succesive sunt codate (într-o fază ulterioară) entropic, cu un algoritm de tip Huffman. Standardul CCITT de transmisie fax recomandă folosirea unor codări Huffman truncheate: astfel, dacă o lungime de secvenţă este mai mică decât 64 este codată direct, cuvântul de cod respectiv numindu-se terminator; dacă o lungime de secvenţă este cuprinsă în gama [64;1791] se codează separat numărul de pixeli ce formează un multiplu de 64 (formând un cod mască, *make-up code*) şi restul împărţirii lungimii secvenţei la 64, cu un cod terminator. În plus, există un cod special de sfârşit de linie (EOL). Tabelele de codare sunt standardizate (se pot găsi de exemplu în [9, pp. 544-545]). Pe lângă transmisia de fax, codarea RLE mai este utilizată în diferite formate de fişiere imagine (BMP, PCX, TGA).

Porţiunea următoare de program C realizează codarea RLE a şirul de intrare *in*, de dimensiune *DIM_IN*, cu cuvinte de cod ce pot reprezenta cel mult *MAX_RUN* simboluri identice succesive; cuvintele de cod sunt scrise în şirul *out*. Poziţiile curente din şirurile de intrare şi ieşire sunt determinate de variabilele *pos_in* şi respectiv *pos_out*. Implementarea presupune că tipul de date al şirului *in* permite aplicarea operatorului de negaţie (*!*).

```
pos_in=0;
out[0]=in[0];
pos_out=1;
crt_value=in[0];
while (pos_in<DIM_IN) {
  if (in[pos_in]==crt_value) then
    if (run_length<MAX_RUN) then
      run_length++;
```

```
    else {
      out[pos_out]=MAX_RUN;
      pos_out++;
      out[pos_out]=0;
      run_length=1; }
  else {
    out[pos_out]=run_length;
    pos_out++;
    run_length=1;
    crt_value=!crt_value; }
  pos_in++; }
```

Codarea WBS

Prima etapă a codării WBS (*White Block Skipping*) presupune împărţirea şirului binar în grupe de câte D simboluri succesive. Principiul codării este simplu: un bloc nul este înlocuit cu un singur simbol de 0, un bloc nenul este prefixat de un simbol de 1 şi copiat. Conform acestei codări (cu grupe de $D = 3$ simboluri), şirul de valori binare $0,0,0,0,1,0,0,0,1,1,1,0,0,0,0$ este împărţit în grupele $(0,0,0)$, $(0,1,0)$, $(0,0,1)$, $(1,1,0)$, $(0,0,0)$ şi codat ca (0), $(1,0,1,0)$, $(1,0,0,1)$, $(1,1,1,0)$, (0), transformându-se în şirul $0,1,0,1,0,1,0,0,1,1,1,1,0,0$.

Dacă şirul dat binar de n grupe de simboluri conţine n_0 grupe nule, în urma compresiei mai rămân $(n-n_0)(D+1)+n_0$ simboluri şi factorul de compresie este atunci:

$$C = \frac{nD}{(n-n_0)(D+1)+n_0} \tag{7.3}$$

Pentru ca să existe compresie trebuie ca $C > 1$ şi deci atunci

$$n_0 D > n \Longleftrightarrow \frac{n_0}{n} > \frac{1}{D} \tag{7.4}$$

adică proporţia de grupe nule din şir trebuie să fie mai mare decât inversul numărului de simboluri dintr-o grupă.

Pentru o largă categorie de imagini, lungimea blocului de codare $D = 10$ poate fi folosită cu rezultate bune [9]. O îmbunătăţire a performanţei se poate obţine dacă se introduce o adaptare a codării, prin detecţia liniilor albe (reprezentate doar cu valori 0) şi codarea unei linii întregi cu un singur 0 (desigur că în acest caz la codurile tuturor blocurilor se mai adaugă un prefix de valoare 1). O altă variantă a metodei exploatează structura bidimensională a imaginii, codând blocuri pătrate din imagine (deci imaginea nu mai este vectorizată).

Porţiunea următoare de program C realizează o variantă de codare WBS cu blocuri de dimensiune D a şirului de intrare *in* (ale cărui valori sunt 0 şi 1) de dimensiune DIM_IN;

codul WBS este scris în şirul de ieşire *out*. Verificarea dacă blocul curent este nul se face prin sumarea valorilor acestuia. Poziţiile curente în şirul de intrare şi şirul de ieşire sunt memorate în *pos_ in* şi respectiv *pos_ out*.

```
pos_in=0;
pos_out=0;
while (pos_in<DIM_IN){
s=0;
for (i=0;i<D;i++)
  s=s+in[pos_in+i];
if (s>0) then {
  out[pos_out]=1;
  pos_out++;
  for (i=0;i<D;i++)
    out[pos_out+i]=in[pos_out+i];
  pos_out=pos_out+D; }
else {
  out[pos_out]=0;
  pos_out++; }
pos_in=pos_in+D; }
```

Codarea Ziv-Lempel

Codarea Ziv-Lempel nu are ca origine ideea explicită de a mări entropia alfabetului codului (precum codarea Huffman), ci se bazează pe tehnica dicţionar LUT (folosită şi la reprezentarea imaginilor indexate): cuvintele de cod asociate unor şiruri de simboluri ale sursei de intrare sunt indicii (numerele de ordine) poziţiilor la care se găsesc respectivele şiruri de simboluri în dicţionarul codului (tabela de codare). Tabela de codare (dicţionarul) este creată pe măsura parcurgerii şirului de simboluri de intrare, şi, pentru decodare, tabela se crează din cuvintele de cod deja transmise (ceea ce înseamnă că nu este necesară memorarea sau transmiterea explicită a tabelului de codare[2]).

Dicţionarul iniţial are două cuvinte: cuvântul 0, având indicele 0 şi cuvântul 1, având indicele 1. Din şirul de intrare (presupus binar) se extrage câte un bit şi se verifică dacă şirul de biţi deja extras se regăseşte sau nu în dicţionar. Dacă şirul se regăseşte în dicţionar, se mai adaugă un bit din şirul de intrare. Dacă şirul nu se regăseşte în dicţionar, atunci şirul este trecut în dicţionar, în şirul codat se scrie codul (indicele) prefixului şirului nou adăugat (deci poziţia la care se găseşte în dicţionar cuvântul la care

[2]Această caracteristică este probabil unică codului Ziv-Lempel; pentru toate celelalte aplicări ale tehnicii LUT, tabela de codare este transmisă sau memorată împreună cu cuvintele de cod sau se presupune existenţa unei tabele de codare implicite, cunoscute (ca de exemplu tabela de culoare cu nivele de gri).

adăugând ultimul bit al şirului de intrare se obţine un cuvânt nou) şi ultimul bit din şir devine primul bit al următorului şir de intrare. Procedeul continuă până când şirul de intrare nu mai are simboluri necodate. Dacă, înaintea terminării simbolurilor din şirul de codat, tabela de codare (cu număr fixat de intrări) se umple, există două variante de continuare: fie restul şirului de intrare se codează conform tabelei de codare existente prin căutarea celor mai lungi cuvinte de cod, fie tabela de codare este "golită" şi se continuă cu procedeul iniţial.

Să considerăm şirul de intrare 1, 0, 0, 0, 0, 1, 0, 1, 1, 1; dicţionarul cuvintelor de cod conţine cuvântul 0 (cu indice 0) şi cuvântul 1 cu indice 1. Simbolul curent este 1 (pe prima poziţie a şirului de intrare) şi formează şirul de biţi extras; acest şir se regăseşte în dicţionar (cu indicele 1) şi atunci se mai extrage un bit din şir; şirul de biţi extraşi devine 10. Cuvântul 10 nu există în dicţionar, şi atunci: în şirul de ieşire se scrie indicele celui mai lung cuvânt din dicţionar la care adăugând un bit obţinem noul cuvânt (noul cuvânt este 10, cuvântul prefix este 1, iar indicele scris la ieşire este 1), în dicţionar se adaugă cuvântul 10 (care va avea indicele 2), iar noua poziţie curentă din şirul de intrare este ultimul bit al şirului deja extras, deci poziţia 2, bitul 0. Simbolul curent este 0; acest şir se regăseşte în dicţionar (cu indicele 0) şi atunci se mai extrage un bit din şir; şirul de biţi extraşi devine 00. Cuvântul 00 nu există în dicţionar, şi atunci: în şirul de ieşire se scrie indicele prefixului noului cuvânt (deci 0), în dicţionar se adaugă cuvântul 00 (care va avea indicele 2), iar noua poziţie curentă din şirul de intrare este ultimul bit al şirului deja extras, deci poziţia 3, bitul 0. Simbolul curent este 0; acest şir se regăseşte în dicţionar (cu indicele 0) şi atunci se mai extrage un bit din şir; şirul de biţi extraşi devine 00. Şirul 00 se regăseşte în dicţionar (indicele 3) şi atunci se mai extrage un bit din şir; şirul de biţi extraşi devine 000. Cuvântul 000 nu există în dicţionar, şi atunci: în şirul de ieşire se scrie indicele prefixului noului cuvânt (deci 3), în dicţionar se adaugă cuvântul 000 (care va avea indicele 4), iar noua poziţie curentă din şirul de intrare este ultimul bit al şirului deja extras, deci poziţia 5, bitul 0. Procedeul continuă în mod analog.

Pentru decodare, fiecare nou cuvânt de cod implică scrierea unei noi intrări în tabelul de codare (dicţionar) în care şirul de simboluri este format din prefix (şirul de simboluri care se găseşte în dicţionar la intrarea precizată de cuvântul de cod) şi o terminaţie de 1 bit, a cărei valoare rezultă la primirea cuvântului de cod următor (să nu uităm că şirurile succesive de simboluri ce se codează au în comun ultimul, respectiv primul bit).

Metoda Ziv-Lempel a pus bazele unei clase mai largi de metode de compresie, incluzând printre altele şi varianta LZW (Lempel-Ziv-Walsh) folosită de utilitarul de compresie ZIP.

7.2 Compresia imaginilor cu nivele de gri

Cea mai imediată metodă de codare a unei imagini cu nivele de gri este de a o considera ca un şir de biţi şi de aplica metodele de codare pentru imagini binare: fie pentru

fiecare plan de bit al reprezentării binare a nivelor de gri, fie pentru succesiunea de biţi a reprezentărilor nivelelor de gri. Asemenea abordări produc codări fără pierderi a imaginilor şi nu produc întotdeauna rezultate spectaculoase. O mult mai mare amploare a căpătat clasa de metode de compresie cu pierderi [controlabile].

7.2.1 Codarea predictivă

Codarea predictivă se bazează pe existenţa unei importante corelaţii spaţiale între valorile pixelilor unei imagini (deci valorile pixelilor vecini sunt asemănătoare). Această corelaţie poate implica, de exemplu, că valoarea unui pixel poate fi aproximată cu o combinaţie a valorilor unora dintre vecinii săi. Dacă se stabileşte o ordine de parcurgere a pixelilor ce formează imaginea (deci dacă se stabileşte o ordine de baleiere a imaginii) şi pentru aproximarea valorii pixelului curent se folosesc pixeli vecini spaţial lui, parcurşi anterior, spunem că prezicem valoarea pixelului curent. Predicţia se realizează printr-o funcţie, care, cunoscută la nivelul întregii imagini permite determinarea unei variante aproximative a imaginii date cunoscând doar un număr mic de "pixeli de start". Pentru a avea o codare cât mai precisă, se folosesc şi erorile dintre valorile prezise şi cele reale; codarea asociată imaginii iniţiale va conţine deci funcţia de predicţie, valorile de start şi erorile de aproximare ale fiecărui punct. Dacă predictorul este determinat în mod corect, atunci eroarea de predicţie $e(n)$ este mică şi reprezentarea ei necesită mult mai puţini biţi decât reprezentarea valorii originale $u(n)$. Schema de codare cu predicţie este reprezentată în figura 7.1.

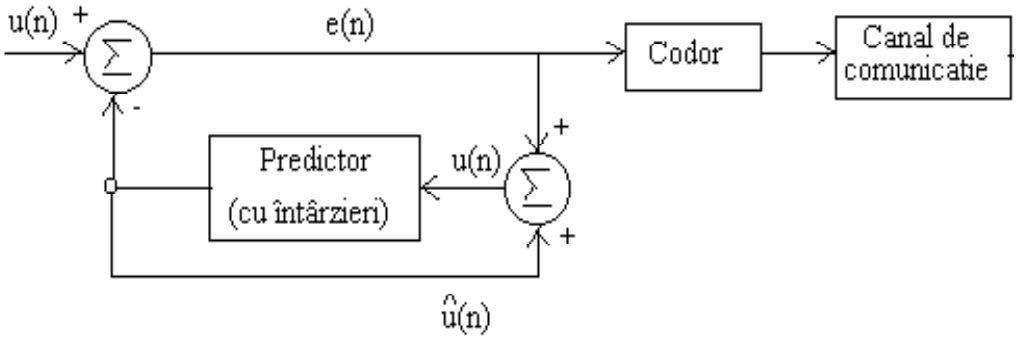

Fig. 7.1: Schema de codare cu predicţie.

Ecuaţiile ce descriu procesul sunt ecuaţia erorii (7.5) (care exprimă eroarea de aproximare $e(n)$ ca diferenţă între valoarea corectă $u(n)$ şi valoarea prezisă $\widehat{u}(n)$) şi ecuaţia de predicţie (7.6) (ce exprimă modul în care se determină valoarea aproximativă $\widehat{u}(n)$ din valorile anterioare $u(n-k_1), u(n-k_2), ...$ pe baza predictorului *pred*):

$$e(n) = u(n) - \widehat{u}(n) \tag{7.5}$$

$$\widehat{u}(n) = pred\,(u(n-k_1), u(n-k_2), ...) \tag{7.6}$$

Procesul de decodare este reprezentat schematic în figura 7.2. Eroarea de predicţie $e_q(n)$ (cuantizată) este adunată la valoarea aproximativă $\widehat{u'}(n)$, determinată cu acelaşi predictor *pred* din valorile $u'(n)$ deja calculate.

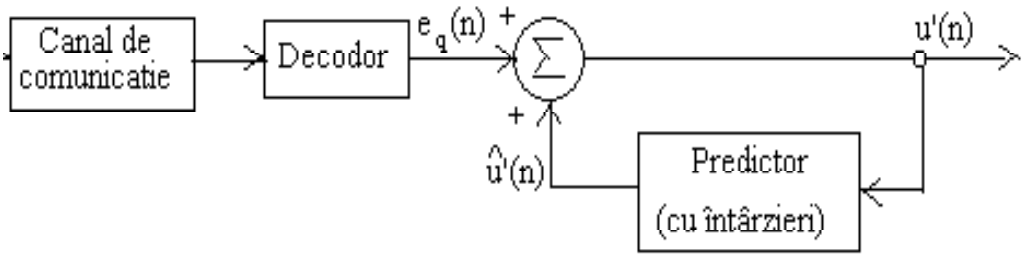

Fig. 7.2: Schema de decodare cu predicţie.

Predictorii cei mai simpli sunt liniari şi sunt invariaţi la schimbarea punctului curent. Predictorii pe linii sau coloană calculează aproximarea în punctul curent $u(m,n)$ al imaginii ca valoarea anterioară de pe aceeaşi linie $\widehat{u}(m,n) = u(m,n-1)$ sau de pe aceeaşi coloană $\widehat{u}(m,n) = u(m-1,n)$ (dacă ordinea de baleiaj este cea uzuală). Se mai pot folosi predictori de tip valoare medie:

$$\widehat{u}(m,n) = \frac{1}{2}\left(u(m-1,n) + u(m,n-1)\right) \tag{7.7}$$

$$\widehat{u}(m,n) = \frac{1}{4}\left(u(m-1,n) + u(m,n-1) + u(m-1,n-1) + u(m-1,n+1)\right) \tag{7.8}$$

Un caz particular de codare cu predicţie este modulaţia delta, caracterizată de cuantizarea erorii de predicţie (aproximare) $e(n)$ cu un singur bit (bit de semn).

7.2.2 Compresia imaginilor cu transformate

Principiul compresiei cu transformate a imaginilor se bazează pe proprietăţile de compactare a energiei şi decorelare a componentelor spectrale pe care le prezintă majoritatea transformărilor integrale unitare (discutate în secţiunea 4.2). Atâta vreme cât cea mai mare parte a energiei este distribuită în câteva componente spectrale (de joasă frecvenţă), toate celelalte pot fi ignorate; astfel memoria necesară reprezentării este mult mai mică. Este evident că o asemenea metodă de compresie este cu pierderi.

Aplicarea practică a compresiei cu transformate trebuie să aibă în vedere trei aspecte: alegerea transformatei, stabilirea frecvenţei limită de la care începe ignorarea valorilor şi, în fine, cuantizarea componentelor spectrale păstrate.

Transformarea optimală trebuie să decoreleze complet componentele spectrale (asigurând astfel şi compactarea maximă a energiei şi cea mai bună eroare de aproximare prin truncherea frecvenţelor înalte). Decorelarea completă este dependentă de proprietăţile statistice ale imaginii (matrice de covariaţie), deci, teoretic, pentru fiecare imagine în parte, trebuie găsită transformarea optimală. Această transformare este transformata Karhunen-Loeve: transformare integrală unitară a cărei matrice de transformare are pe coloane vectorii proprii normaţi ai matricii de covariaţie a imaginii. Cum această transformare este evident unică pentru o clasă de imagini, în practică se încearcă găsirea unei aproximaţii. În condiţiile în care majoritatea imaginilor naturale pot fi aproximate printr-un model Markov puternic corelat (exprimând dependenţa puternică a valorii pixelilor de valorile vecinilor lor imediaţi), transformata cosinus s-a dovedit o foarte bună alegere.

Cuantizarea componentelor spectrale poate integra şi selecţia componentelor cel mai importante: componentele de frecvenţă joasă sunt cuantizate o precizie mai mare, iar componentele de frecvenţă înaltă sunt cuantizate grosier (echivalent chiar cu eliminarea acestora). Numărul de nivele de cuantizare şi distribuţia acestora (diferenţele dintre nivelele vecine) este adaptate statisticii semnalului (cuantizarea optimă este cuantizarea Loyd-Max [9], [16]).

Exemplul cel mai des folosit de compresie cu transformate este standardul de compresie JPEG (fişiere imagine cu extensia .jpg). Imaginea este divizată în blocuri de 8 x 8 pixeli, care nu se suprapun. Fiecărui bloc i se aplică o transformată cosinus bidimensională, iar cei 64 de coeficienţi ai transformării sunt copiaţi într-un vector prin baleierea pe diagonală a blocului de 8 x 8 pixeli. Coeficienţii sunt cuantizaţi în conformitate cu un număr prestabilit de nivele de cuantizare (stabilit prin standard, şi proporţional cu factorul de calitate dorit pentru imaginea refăcută). Coeficienţii corespunzând frecvenţelor nule (valorile medii ale blocurilor) sunt codate predictiv printr-o tehnică de tip DPCM (*Differential Pulse Code Modulation*). Valorile celorlalţi coeficienţi sunt codaţi entropic (Huffman). Factorii de compresie ce rezultă sunt cuprinşi în mod tipic între 10 şi 100. Aspectul de compresie cu pierderi (diferenţele faţă de imaginea originală) se manifestă prin efectul de *blocking*: sublinierea frontierelor de separaţie a blocurilor de bază (efect observabil şi în figura 7.3).

7.2.3 Codarea cu arbori cuaternari

Un arbore cuaternar (numit în engleza *quadtree*) este un arbore în care fiecare nod neterminal are exact patru descendenţi.

Orice imagine pătrată, de dimensiune putere a lui 2 ($N = 2^K$) poate fi reprezentată

Fig. 7.3: Imagine decodată în urma unei compresii JPEG cu raport de compresie 23 (factor de calitate 90)

(într-o reprezentare de tip ierarhic) pe o structură de arbore cuaternar. Nodurile de pe fiecare nivel al arborelui corespund unei împărţiri a unei zone pătrate din imagine în patru "sferturi". Rădăcina arborelui este asociată întregii imagini (imaginii iniţiale), nodurile de pe primul nivel al arborelui corespund celor patru sferturi ale imaginii, nodurile de pe nivelul doi corespund sferturilor fiecărui sfert anterior determinat al imaginii, şi aşa mai departe. Împărţirea imaginii poate continua până când nodurile nivelului curent al arborelui corespund unor zone pătrate a căror dimensiune este de un pixel. Adâncimea arborelui astfel obţinut este K, şi fiecare pixel al imagini va corespunde unui nod terminal (frunză) de pe ultimul nivel al arborelui. Fiecare nod terminal conţine informaţia de valoare a pixelului la care este asociat.

Structura arborelui anterior poate fi simplificată prin introducerea în etapa de construcţie a unui test de uniformitate a regiunilor reprezentate de fiecare nod: dacă regiunea pătrată considerată nu este uniformă, atunci aceasta va fi descompusă prin tăiere în patru părţi egale şi nodul corespunzător va deveni neterminal (va "căpăta" cei patru descendenţi). Dacă regiunea pătrată considerată este uniformă (deci este compusă din pixeli de acelaşi fel), nodul respectiv devine un nod frunză (terminal) al arborelui (deci nu mai are descendenţi). Fiecare nod terminal conţine informaţia de valoare a zonei de imagine la care este asociat (vezi figura 7.4). O zonă este considerată uniformă dacă diferenţa maximă de nivel de gri a pixelilor ce o formează nu depăşeşte un anumit prag impus; valoarea zonei uniforme este media nivelelor de gri a pixelilor ce o compun.

Pentru refacerea imaginii inţiale din reprezentarea arborescentă este suficientă alegerea nodurilor terminale a căror valoare corespunde pixelilor de obiect. Adâncimea la care este plasată în arbore o frunză conţine informaţia de dimensiune a zonei pătrate corespunzătoare din imagine (o frunză situată la adâncimea h corespunde unei zone pătrate de latură 2^{K-h} pixeli). Poziţia frunzei faţă de nodurile de pe acelaşi nivel ce au acelaşi

predecesor este direct determinată de regula de alocare a sferturilor unei zone la nodurile descendente ale arborelui (regula de alocare trebuie să se păstreze pentru întregul arbore) (vezi figura 7.5).

Codarea imaginii (sau a arborelui cuaternar asociat) se face prin memorarea poziţiei în arbore a nodurilor terminale şi a valorilor acestora. Poziţia în arbore a unui nod se specifică prin descrierea căii prin care se ajunge la acesta, pornind de la rădăcina arborelui; această cale va conţine codurile de alocare a descendenţilor ce corespund avansului în adâncime în arbore.

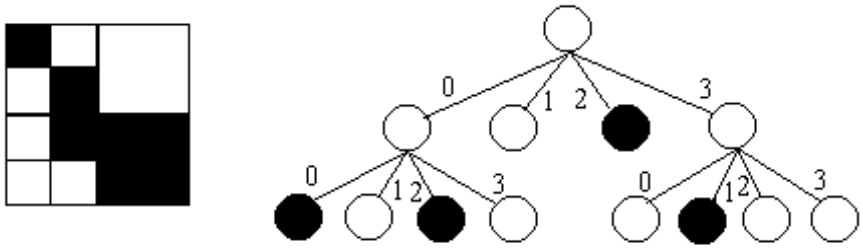

Fig. 7.4: Exemplu de reprezentare a unei imagini binare pe un arbore cuaternar complet

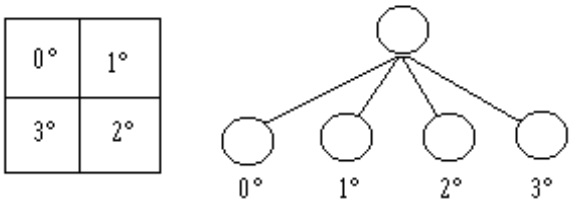

Fig. 7.5: Regula de alocare a descendenţilor

Principalul inconvenient al metodei de etichetare folosind arborele cuaternar este legat de complexitatea construirii acestuia prin abordarea ierarhică *top-down* (de sus în jos) prezentată; în particular, testul de uniformitate la nivelul fiecărui bloc presupune testarea valorilor tuturor pixelilor care compun blocul. Pe ansamblu, aceasta duce la parcurgerea fiecărui pixel din imagine de un număr de ori egal cu adâncimea în arborele cuaternar al blocului pătrat din care face parte.

Pentru a înlătura acest incovenient este suficient ca pixelii imaginii să nu mai fie parcurşi în ordinea tradiţională de baleiaj (pe linii, de la stânga la dreapta şi de sus în jos), ci

într-o altă ordine, care să îi prearanjaze pe grupuri ce corespund pătratelor de diviziune a imaginii. Un asemenea baleiaj este reprezentat de o curbă de umplere a spaţiului: un parcurs ce trece o singură dată prin fiecare pixel al imaginii, nu se autointersectează şi în care oricare doi pixeli parcurşi consecutiv sunt vecini spaţial în imagine (într-o vecinătate de tip V_4 sau V_8). Curbele de umplere a spaţiului sunt structuri fractale, definite prin repetarea la diferite nivele ierarhice a unei aceleiaşi structuri. Pentru baleiajul imaginilor s-au reţinut două astfel de curbe: curba Peano-Hilbert (numită şi curba Peano în U, după forma celulei sale de bază) (vezi figura 7.6) şi curba Morton (sau curba Peano în Z) (vezi figura 7.7).

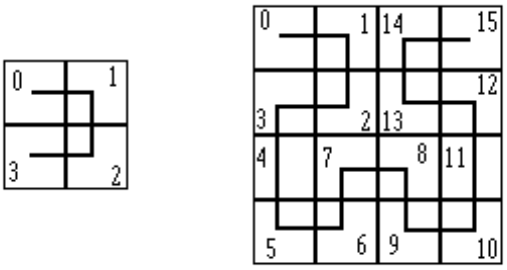

Fig. 7.6: Ordinea de parcurgere a pixelilor pentru curba Peano în U, la două nivele de rezoluţie

Dispunând de o astfel de ordine de baleiere, este evident că dacă se parcurge imaginea în acestă ordine, zonele pătrate uniforme (cu pixelii de aceeaşi valoare) sunt detectate într-o singură trecere şi astfel arborele cuaternar poate fi creat direct prin nodurile sale terminale. Pentru o implementare eficientă este însă necesară şi deducerea rapidă a indicelui pe curba de baleiere a pixelilor, pornind de la coordonatele lor în imagine. Doar curba Peano în Z are o asemenea relaţie rapidă de calcul, prin întreţeserea biţilor ce dau coordonatele în imagine a punctului. Cuvântul binar ce exprimă indicele pe curbă a oricărui punct este format din biţii din coordonata verticală, ce vor ocupa poziţiile de ordin par şi din biţii din coordonata orizontală ce vor ocupa poziţiile de ordin impar (păstrându-şi aceeaşi ordine de rang).

7.2.4 Cuantizarea vectorială

Cuantizarea vectorială este un algoritm de compresie a imaginilor ce se aplică asupra unor date vectoriale şi nu scalare, putând fi interpretat ca o extensie a conceptului de cuantizare scalară. Cuantizarea scalară asociază unei mulţimi mari de valori numere dintr-o mulţime mai mică (în mod tipic acestea din urmă fiind numere naturale); asocierea include (chiar dacă nu explicit) operaţii de tipul rotunjirii la cel mai apropiat întreg. Cuantizarea vectorială aproximează (sau rotunjeşte) un grup de numere deodată, nu doar unul singur. Aşadar, pentru a realiza o cuantizare vectorială sunt necesare un set

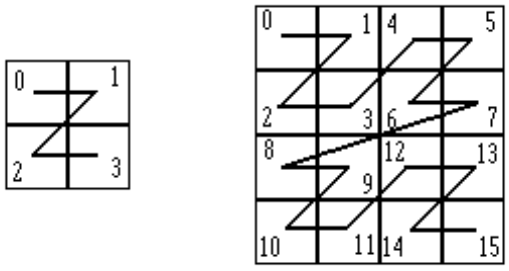

Fig. 7.7: Ordinea de parcurgere a pixelilor pentru curba Peano în Z, la două nivele de rezoluţie

de vectori de aproximare (inclusiv metoda prin care acestea pot fi deduse) şi o regulă de asociere a vectorilor de intrare cu vectorii de aproximare.

Să notăm cu $\mathbf{x}_i$ al i-lea vector de intrare şi cu $\boldsymbol{\mu}_j$ al j-lea vector de aproximare. Operaţia de cuantizare presupune înlocuirea vectorilor de intrare $\mathbf{x}_i$ cu vectori de aproximare $\boldsymbol{\mu}_j$, introducând deci erori; pentru ca erorile (măsurate de eroarea pătratică medie) să fie cât mai mici, este necesar ca pentru fiecare vector de intrare, aproximarea să se facă cu vectorul de aproximare cel mai apropiat (în sensul distanţei euclidiene). Aceasta este regula de asociere. Dacă există n vectori de aproximare ce pot fi folosiţi, aceştia se pot grupa într-un tabel de codare (existent şi la codare şi la decodare), iar fiecare vector de aproximare $\boldsymbol{\mu}_j$ va fi reprezentat doar prin indicele j (deci o altă aplicare a tehnicii LUT). Dacă vectorii de intrare au p componente, codate fiecare cu câte b biţi, iar numărul de vectori de aproximare poate fi reprezentat pe n_b biţi, atunci factorul de compresie realizat de cuantizarea vectorială este dat de[3]:

$$C = \frac{pb}{n_b} \tag{7.9}$$

Pentru cazul imaginilor, vectorii de intrare se aleg ca blocuri pătrate, nesuprapuse între ele, din imagine. Dimensiuni uzuale ale acestor blocuri sunt 4 x 4 şi 8 x 8 (rezultând deci vectori de intrare cu 16, respectiv 64 de componente). Dacă considerăm cazul imaginilor cu nivele de gri uzuale, reprezentate cu 256 nivele de gri ($b = 8$) şi dimensiuni ale tabelei de codare $n = 256$ (256 vectori de aproximare), atunci raportul de compresie este de 16, respectiv 64.

Construirea tabelei de codare (determinarea vectorilor de aproximare) se realizează în mod clasic prin algoritmi de clustering iterativ. În original (adică în limba engleză) termenul de "cluster" defineşte un grup, ciorchine, snop sau o clasă de unităţi, "asemănătoare". Asemănarea unităţilor este determinată în mod uzual prin asociere, similaritate sau distanţă între unităţi (vectori). Algoritmul de clustering este procesul prin care unei

[3]Acest mod de calcul al raportului de compresie nu ţine seama de necesitatea transmiterii sau memorării şi a tabelului de codare, de dimensiune npb biţi.

mulţimi de unităţi (entităţi) i se asociază, element cu element, o informaţie de apartenenţă la un anumit grup. Mai general, putem interpreta procesul de clustering ca un proces de partiţionare a unui set de unităţi într-un număr de submulţimi (numite clase sau clustere), pe baza unui anumit criteriu. Indiferent de criteriul folosit, se doreşte obţinerea unor clustere distincte, omogene şi bine separate. Numărul de clustere în care se face împărţirea setului de unităţi nu este în mod obligatoriu cunoscut apriori.

Metodele de clustering iterativ distribuie obiectele (vectorii) într-un număr predefinit de clase, repetând testarea unor condiţii pentru fiecare obiect al mulţimii; în funcţie de îndeplinirea sau nu a respectivelor condiţii, partiţia existentă la un moment dat este declarată corespunzătoare sau, în urma modificării alocării unor unităţi, procedeul de verificare se reia. Se poate considera că algoritmii iterativi fac mai multe "treceri" prin setul de obiecte de partiţionat, până la obţinerea stabilizării (convergenţei) valorii criteriului ce caracterizează calitatea partiţiei.

Calitatea partiţiei (a clasificării) este măsurată de suma varianţelor clusterelor (adică suma distanţelor de la fiecare vector la centrul clasei în care aparţine, ceea ce poate fi interpretat şi ca o eroare de aproximare a vectorilor din clase prin centrul respectivei clase). Deci funcţia criteriu ce trebuie minimizată este

$$J = \sum_{j=1}^{n} J_j = \sum_{j=1}^{n} \sum_{\mathbf{x}_i \in \omega_j} \left\| \mathbf{x}_i - \boldsymbol{\mu}_j \right\|^2 = \sum_{j=1}^{n} \sum_{i=1}^{N} u_{ij} \left\| \mathbf{x}_i - \boldsymbol{\mu}_j \right\|^2 \tag{7.10}$$

În urma minimizării lui J trebuie determinate valorile $\boldsymbol{\mu}_j$ (centrele claselor) şi valorile binare u_{ij}, coeficienţii de apartenenţa ai vectorilor i la clasele j, definiţi de:

$$u_{ij} = \begin{cases} 1, \text{ dacă } \mathbf{x}_i \in \omega_j \\ 0, \text{ dacă } \mathbf{x}_i \notin \omega_j \end{cases} \tag{7.11}$$

Determinarea centrelor claselor se poate face simplu, prin anularea derivatei în raport cu $\boldsymbol{\mu}_j$ a funcţiei criteriu J

$$\frac{\partial J}{\partial \boldsymbol{\mu}_j} = 2 \sum_{i=1}^{N} u_{ij} \left(\boldsymbol{\mu}_j - \mathbf{x}_i \right) = 0 \tag{7.12}$$

de unde rezultă că centrele claselor sunt mediile vectorilor ce aparţin acestora:

$$\boldsymbol{\mu}_j = \frac{\sum\limits_{i=1}^{N} u_{ij} \mathbf{x}_i}{\sum\limits_{i=1}^{N} u_{ij}} \tag{7.13}$$

Determinarea coeficienţilor de apartenenţă u_{ij} este însă o problemă de optimizare combinatorială, care nu poate fi rezolvată analitic. Pentru rezolvarea acestei probleme sunt

necesare metode iterative. Metoda imediată urmăreşte să determine, pentru fiecare vector al setului, dacă acesta poate fi mutat dintr-o clasă în alta, astfel ca suma varianţelor claselor să scadă. După fiecare asemenea mutare, este necesară actualizarea mediilor claselor între care s-a făcut schimbul. Iteraţiile se repetă până când nici un vector nu mai poate fi mutat. Algoritmul poate fi descris în etapele următoare:

1. se alege o partiţie aleatoare a setului de obiecte (vectori)
2. pentru fiecare vector $\mathbf{x}_i$ din set, dacă nu este unic în clasa sa ω_j, se calculează costul mutării în altă clasă, ω_k, $k \neq j$; acest cost este
$$c_k = \frac{n_k}{n_k + 1} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2 - \frac{n_j}{n_j - 1} \left\|\mathbf{x}_i - \boldsymbol{\mu}_j\right\|^2 \tag{7.14}$$
3. vectorul $\mathbf{x}_i$ este mutat în clasa pentru care costul c_k este minim; se recalculează mediile claselor implicate în schimbare (ω_j şi ω_k)
4. dacă cel puţin un vector a fost mutat între două clase, algoritmul se reia de la pasul 2.

Principalul dezavantaj al acestei abordări este faptul că mediile claselor sunt recalculate după fiecare schimbare ce implică fiecare vector al mulţimii considerate, ceea ce are ca efect un volum mare de calcule. O simplificare a metodei provine din observaţia intuitivă că este normal ca un obiect să fie alocat (să aparţină) clasei de care este cel mai apropiat (în sensul distanţei la media acesteia). Folosind acestă observaţie, realocarea se poate face pentru toate obiectele considerate fără a fi nevoie de recalcularea mediilor claselor pentru fiecare obiect; recalcularea mediilor se va face după fiecare parcurgere completă a mulţimii de obiecte de partiţionat. Acest algoritm este algoritmul "Basic ISODATA[4]" (cunoscut şi sub numele de *k-means* - "cele k medii"). Algoritmul poate fi descris de următoarele etape:

1. se alege o partiţie aleatoare a setului de obiecte (vectori)
2. pentru fiecare vector $\mathbf{x}_i$ din set, se calulează distanţele sale la mediile tuturor claselor,
$$d_k = \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2 \tag{7.15}$$
3. în iteraţia următoare, vectorul $\mathbf{x}_i$ va fi mutat în clasa la care distanţa d_k este minimă
4. după parcurgerea completă a setului de vectori, se reactualizează mediile claselor

[4]ISODATA este acronimul de la "Iterative Self Organizing Data Analysis Technique" - tehnică iterativă cu auto-organizare de analiză a datelor, apărut prin 1965.

5. dacă (faţă de iteraţia anterioară) nici un vector nu a fost mutat în altă clasă (sau media nici unei clase nu s-a modificat), algoritmul se încheie; dacă nu, se reia algoritmul de la pasul 2.

Pentru nici unul dintre algoritmii iterativi prezentaţi nu se poate preciza numărul de parcurgeri ale setului de vectori (obiecte) de partiţionat. Numărul de iteraţii este puternic dependent de alegerea partiţiei iniţiale a vectorilor, precum şi de organizarea intrinsecă a acestora.

Figura 7.8 prezintă o imagine refăcută după o compresie prin cuantizare vectorială, cu raport mare de compresie (128). Se remarcă slaba calitate a imaginii refăcute, cauza fiind numărul mic (16) de vectori de cod folosiţi. În imagine sunt foarte vizibile frontierele dintre blocurile de 8 x 8 folosite.

Fig. 7.8: Imagine refăcută după codarea prin cuantizare vectorială; codarea a fost realizată cu blocuri de 8 x 8 pixeli; tabelul vectorilor de cod are 16 intrări, deci se obţine o compresie de 128.

Capitolul 8

SEGMENTAREA IMAGINILOR

Segmentarea imaginilor se referă la descompunerea unei scene (imagini) în componentele sale [9]. În urma procesului de segmentare vor fi extrase din imagine obiecte distincte, regiuni ce satisfac anumite criterii de uniformitate, sau alte elemente.

În [19] se propune o definiţie matematizată a procesului de segmentare, şi anume segmentarea unei imagini f este definită ca partiţionarea [completă] a lui f (8.1) într-un ansamblu de mulţimi disjuncte nevide şi conexe (8.2), ce satisfac fiecare un anumit criteriu C (8.3), criteriu ce nu mai este respectat pentru reuniunea oricăror două elemente ale partiţiei.

$$f = \bigcup_{i=1}^{C} f_i \ , \quad f_i \text{ conexe} \tag{8.1}$$

$$f_i \bigcap f_j = \emptyset, \forall i \neq j \text{ şi } f_i \text{ este conexă}, \forall i \tag{8.2}$$

$$\mathsf{C}(f_i) = TRUE, \forall i \text{ şi } \mathsf{C}\left(f_i \bigcup f_j\right) = FALSE, \forall i \neq j \tag{8.3}$$

Alegerea unei tehnici specifice de segmentare (partiţionare a imaginii) este legată de mai multe aspecte caracteristice imaginii de analizat şi cerinţelor utilizatorului. După natura şi conţinutul imaginii, tehnicile de segmentare trebuie să ţină cont de prezenţa în imagine a diverse categorii de artefacte:

- reflexii, iluminare neomogenă
- zgomot suprapus infomaţiei utile

- zone texturate

După primitivele de extras, tehnicile de segmentare se împart în două categorii fundamentale: tehnicile de segmentare orientate pe regiuni şi tehnicile de segmentare orientate pe contur. Primitivele extrase din imagine sunt regiuni (forme) şi zone texturate pentru tehnicile orientate pe regiuni, sau entităţi de tip discontinuitate (frontiere, segmente de dreaptă, unghiuri) pentru tehnicile orientate pe contur. În cadrul segmentării orientate pe regiuni se disting câteva categorii principale de tehnici:

- etichetarea imaginilor binare
- segmentarea pe histogramă
- creşterea şi fuziunea regiunilor
- segmentarea texturilor
- segmentarea prin metode de clustering

Tehnicile principale de segmentare orientată pe contururi sunt:

- extragerea contururilor prin metode de gradient şi derivative
- extragerea contururilor prin metode neliniare
- extragerea contururilor prin metode liniare optimale
- extragerea contururilor prin modelare matematică

În cele ce urmează se prezintă doar o parte dintre aceste tehnici, pe care le considerăm cele mai semnificative.

8.1 Segmentarea orientată pe regiuni

8.1.1 Segmentarea bazată pe histogramă

În general, operaţia de segmentare orientată pe regiuni urmăreşte extragerea din imagine a zonelor (regiunilor) ocupate de diferitele obiecte prezente în scenă. Un obiect se defineşte ca o entitate caracterizată de un set de parametri ale căror valori nu se modifică în

diferitele puncte ce aparţin entităţii considerate. Mai simplu, putem spune că obiectul are proprietatea de uniformitate a parametrilor de definiţie.

Unul dintre cei mai simpli parametri de definiţie este nivelul de gri al punctului. Nivelul de gri corespunde în scenă unei proprietăţi fizice [9] (reflectanţă, transmitivitate, valoare tristimulus, etc.) ce este preluat de senzorul de imagine şi asociat luminanţei imaginii. În acest caz, histograma imaginii (funcţia de densitate de probabilitate a variabilei aleatoare discrete ale cărei realizări sunt nivelele de gri din imagine) reflectă distribuţia în scenă a proprietăţii fizice înregistrate. Pentru o imagine f de $M \times N$ pixeli şi L nivele de gri, histograma este definită (8.4) ca probabilitatea de apariţie în imagine a diferitelor nivele de gri posibile.

$$h(i) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \delta(i - f(m,n)) \quad , \quad i = 0, 1, ...L-1 \tag{8.4}$$

Dacă nivelul de gri (respectiv proprietatea fizică pe care acesta o reprezintă) caracterizează în mod suficient obiectele din scenă, histograma imaginii va prezenta o structură de moduri dominante - intervale de nivele de gri ce apar cu probabilitate mai mare. Fiecare asemenea mod (maxim al histogramei) va reprezenta o anumită categorie de obiecte.

Ca exemplu imediat se poate cita cazul imaginilor obţinute prin scanarea documentelor scrise şi a tipăriturilor sau imaginile în infraroşu (temperatura punctelor este asociată nivelelor de gri astfel încât mai fierbinte însemnă mai alb). Pentru toate aceste tipuri de imagini histograma este de tipul celei prezentate în figura 8.1.

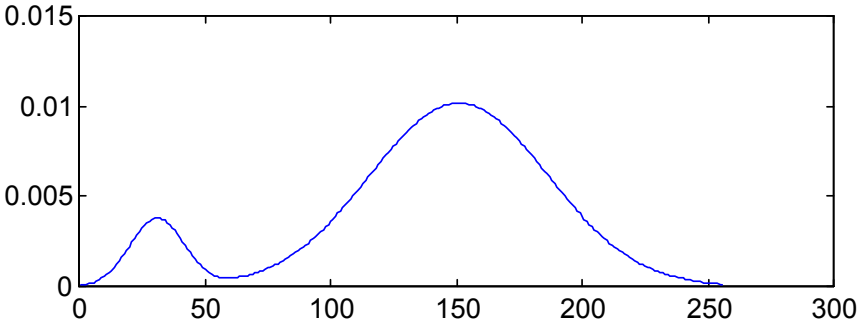

Fig. 8.1: Histogramă bimodală

Tehnici de prăguire (*thresholding*)

Separarea modurilor histogramei (şi deci identificarea obiectelor din imagine, respectiv caractere scrise / pagini albe şi obiecte fierbinţi / obiecte reci) se face prin alegerea unui

nivel de gri T, numit prag de segmentare. Acest prag de segmentare se alege pe minimul global al histogramei. Din imaginea iniţială f de nivele de gri se construieşte o imagine de etichete (imagine etichetată) g, conform transformării descrise de (8.5) (vezi figura 8.2).

$$g(m,n) = \begin{cases} E_0, 0 \leq f(m,n) < T \\ E_1, T \leq f(m,n) < L \end{cases} \tag{8.5}$$

Imaginea etichetată va fi descrisă de două etichete: E_0 pentru punctele al căror nivel de gri este mai mic decât pragul T şi E_1 pentru punctele al căror nivel de gri este mai mare decât pragul T. Etichetele E_0 şi E_1 pot fi valori numerice (0 şi 1, sau 0 şi 255) sau pot fi şiruri de simboluri sau alţi identificatori.

Fig. 8.2: Transformări punctuale de binarizare.

Transformarea (8.5) este o transformare punctuală (noua valoare din punctul (m,n) depinde doar de valoarea anterioară din punctul (m,n)) şi poartă numele de binarizare. Această denumire provine din faptul că rezultatul transformării (imaginea etichetată) este o imagine binară - deci o imagine caracterizată doar de două valori. Se poate remarca de asemenea faptul că binarizarea este un caz particular al transformării de modificare liniară a contrastului (2.2), în care limitele domeniilor de contrastare sunt egale ($T_1 = T_2$) şi contrastarea se face la valorile limită ale nivelelor de gri ($\alpha = 0$, $\beta = L - 1$)[1].

Segmentarea pe histogramă (numită şi prăguire sau *thresholding*) semnifică determinarea unor nivele de gri ce separă modurile histogramei. Tuturor punctelor din imagine al căror nivel de gri corespunde unui acelaşi mod, li se asociază o aceeaşi etichetă (număr, şir de simboluri), rezultând o imagine etichetată, ce pune în evidenţă diferitele obiecte ale scenei iniţiale.

[1] Există însă şi o variantă de binarizare cu două praguri (transformare punctuală numită decupare - "slicing"), (figura 8.2) definită de ecuaţia următoare:

$$g(m,n) = \begin{cases} E_0, \text{ dacă } f(m,n) < T_1 \text{ sau } f(m,n) > T_2 \\ E_1, \text{ în rest} \end{cases}$$

În cazul general al existenţei a mai multe praguri de segmentare T_k, transformarea de segmentare pe histogramă este descrisă de (8.6)

$$g(m,n) = E_k \quad \text{dacă} \quad T_k \leq f(m,n) < T_{k+1} \tag{8.6}$$

unde $T_0 = 0$, $T_C = L$, $k = 0, 1, ..., C-1$.

Pragurile T_k se aleg prin inspecţia histogramei, în minimele locale ale acesteia. Acest tip de segmentare multinivel este mai puţin eficient decât binarizarea, din cauza dificultăţii de stabilire a pragurilor care să izoleze eficient intervalele de interes din histogramă, mai ales atunci când numărul modurilor este mare. Trebuie de asemenea remarcat faptul că este necesară cunoaşterea numărului de tipuri de obiecte din imagine, pentru alegerea corespunzătoare a numărului de praguri de segmentare. În marea majoritate a cazurilor, segmentarea obţinută nu este corectă (există regiuni prost etichetate); ca o regulă generală de îmbunătăţire a performanţelor, se recomandă aplicarea, înaintea segmentării, a unor operaţii de filtrare (eliminare a zgomotului), contrastare, îmbunătăţire, netezire a histogramei - numite preprocesări.

În general, se admite clasificarea metodelor de segmentare pe histogramă [5] după atributele global, local şi dinamic. Aceste atribute se referă la modul de calcul al pragurilor de segmentare T_k în funcţie de nivelul de gri din fiecare punct al imaginii $f(m,n)$, coordonatele punctelor din imagine (m,n) şi o anumită proprietate locală $p(m,n)$ a punctului (m,n), conform (8.7):

$$T_k = T_k(f(m,n), p(m,n), (m,n)) \tag{8.7}$$

Segmentarea se numeşte globală dacă pragurile depind doar de nivelele de gri ale punctelor imaginii:

$$T_k = T_k(f(m,n)) \tag{8.8}$$

Segmentarea multinivel descrisă de (8.6) este în mod evident o metodă de tip global.

Segmentarea se numeşte locală dacă pragurile depind de nivelul de gri şi de anumite atribute locale calculate pentru vecinătăţi ale fiecărui punct:

$$T_k = T_k(f(m,n), p(m,n)) \tag{8.9}$$

Segmentarea se numeşte dinamică dacă pragurile depind de poziţionarea punctelor în imagine (forma cea mai generală a modului de deducere pragurilor) (8.7).

Determinarea automată a pragurilor: metoda Bhattacharya

Metoda Bhattacharyya se bazează pe descompunerea histogramei în moduri individuale Gaussiene, adică se încearcă exprimarea histogramei imaginii ca o sumă ponderată de funcţii de densitate de probabilitate de tip normal (Gaussian). Modelarea modurilor histogramei imaginilor prin distribuţii normale este o presupunere ce se întâlneşte în multe tehnici de prelucrare şi analiză şi pare a fi justificată de considerarea imaginii ca provenind dintr-o imagine ideală, în care fiecare tip de obiect este reprezentat de un unic nivel de gri, peste care s-a suprapus un zgomot alb, aditiv, gaussian. În acest mod, mediile modurilor din histogramă corespund nivelelor de gri ce caracterizează obiectele scenei, iar varianţele acestor moduri sunt determinate de zgomotul suprapus imaginii (care nu este obligatoriu să afecteze în acelaşi mod toate nivelele de gri).

Pentru segmentarea după metoda Bhattacharyya nu este necesară precizarea unui număr de clase (praguri de segmentare), acesta urmând a fi determinat în mod automat. Ideea de plecare a metodei este de a determina parametrii caracteristici ai unei distribuţii normale. Pentru o distribuţie normală

$$N(\mu_k, \sigma_k)(x) = \frac{1}{\sigma_k\sqrt{2\pi}} e^{-\frac{(x-\mu_k)^2}{2\sigma_k^2}}$$

derivata logaritmului este:

$$\frac{\delta \ln N(\mu_k, \sigma_k)(x)}{\delta x} = -\frac{x}{\sigma_k^2} + \frac{\mu_k}{\sigma_k^2} = m_k x + n_k \tag{8.10}$$

Se observă prin examinarea expresiei (8.10) că derivata logaritmului distribuţiei normale este o dreaptă de pantă negativă, din ai cărei parametri se pot deduce media şi varianţa distribuţiei. Parametrii statistici ai distributiei sunt daţi de ecuaţiile (8.11).

$$\sigma_k = \sqrt{\frac{1}{|m_k|}}, \text{şi } \mu_k = \frac{n_k}{|m_k|} \tag{8.11}$$

Această observaţie poate fi aplicată şi pentru o mixtură de distribuţii normale. Să considerăm că histograma h a imaginii este compusă prin superpoziţia aditivă a C moduri gaussiene $N(\mu_k, \sigma_k)$, adică:

$$h(x) = \sum_{k=1}^{C} w_k N(\mu_k, \sigma_k)(x)$$

Din parametrii dreptei se pot determina deci conform (8.11) parametrii statistici ai distribuţiei locale.

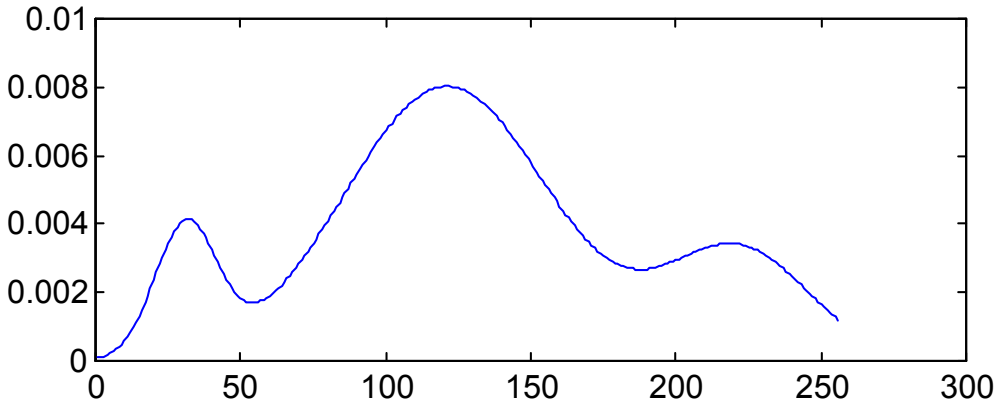

Fig. 8.3: Histogramă cu trei moduri normale

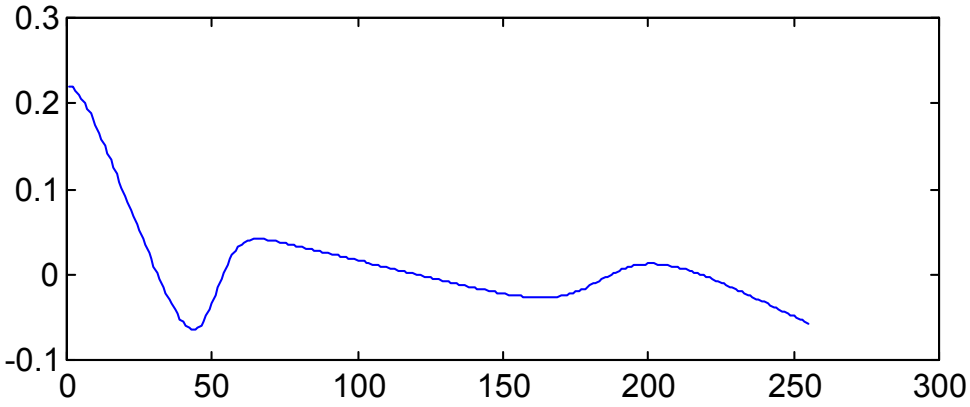

Fig. 8.4: Aplicarea metodei Bhattacharyya pentru histograma trimodală prezentată anterior; se pot observa intervalele pe care funcţia este liniară şi descrescătoare, ce corespund modurilor.

Aşadar, pentru aplicarea metodei la segmentarea pe histogramă a imaginilor, se va studia comportamentul derivatei logaritmului histogramei, adică a funcţiei $z(a)$:

$$z(a) = \ln \frac{h(a)}{h(a-1)}, a = \overline{1, L-1} \tag{8.12}$$

Pentru funcţia astfel construită, se determină intervalele pe care acesta este descrescătoare (vezi figura 8.4); limitele superioare ale acestor intervale sunt pragurile T_k de segmentare pe histogramă. Suplimentar, pe fiecare dintre aceste intervale se poate face o aproximare liniară a punctelor şi pe baza parametrilor deduşi pentru dreapta de aproximare se pot calcula, conform (8.11) parametrii statistici locali.

Principalele inconveniente ale metodei derivă din faptul că presupunerea alcătuirii histogramei imaginii numai din moduri gaussiene nu este întotdeauna adevărată. Ca rezultat, metoda Bhattacharrya va identifica un număr mai mare de praguri decât este necesar, producând fenomenul de suprasegmentare.

Segmentarea cu prag optim

Metoda de segmentare cu prag optim [5], [3], [19] face apel la teoria deciziilor (criteriul de decizie Bayes) pentru stabilirea valorii pragurilor de segmentare ce optimizează un anumit criteriu de eroare. Informaţiile apriori necesare pentru aplicarea unei asemenea tehnici sunt numărul de tipuri de obiecte din imagine, C, procentele de ocupare a imaginii de către fiecare tip de obiecte, P_i şi distribuţia nivelelor de gri ce caracterizează fiecare tip de obiect, $p_i(x)$. Atunci histograma imaginii va fi determinată de mixtura distribuţiilor tipurilor de obiecte:

$$h(x) = \sum_{i=1}^{C} P_i p_i(x), \sum_{i=1}^{C} P_i = 1 \tag{8.13}$$

Cazul cel mai simplu şi mai des folosit este cel al binarizării (8.5), în care trebuie determinat un unic prag T ce separă distribuţiile celor două tipuri de obiecte din imagine (în mod tipic, obiecte "utile" şi fundal). Criteriul ce se urmăreşte optimizat este eroarea de segmentare (clasificare) a punctelor din imagine, adică este dat de numărul de pixeli ce aparţin primului tip de obiect, dar au nivelul de gri mai mare ca pragul T (fiind deci alocaţi greşit celui de-al doilea tip de obiect) şi numărul de pixeli ce aparţin celui de-al doilea tip de obiect, dar au nivelul de gri mai mic decât pragul de segmentare T (fiind deci alocaţi greşit primului tip de obiect). Aşadar, eroarea de segmentare va fi dată de (8.14):

$$E(T) = P_1 \int_{T}^{+\infty} p_1(x)dx + P_2 \int_{-\infty}^{T} p_2(x)dx \tag{8.14}$$

Pragul optim va minimiza eroarea de segmentare a pixelilor. Minimizarea erorii (8.14) conduce la rezolvarea ecuaţiei (8.15), în necunoscuta T.

$$\frac{\partial E(T)}{\partial T} = 0 \tag{8.15}$$

Derivând (8.14) se obţine forma echivalentă a ecuaţiei (8.15):

$$P_1 p_1(T) = P_2 p_2(T) \tag{8.16}$$

După cum am menţionat şi în secţiunea dedicată tehnicilor de segmentare ce nu folosesc informaţii apriori despre imagine (metoda Bhattacharyya), presupunerea că distribuţia nivelelor de gri a diferitelor tipuri de obiecte este de tip normal (Gaussian) este relativ des întâlnită. În aceste condiţii, distribuţiile $p_1(x)$ şi $p_2(x)$ sunt distribuţii normale, $N_1(\mu_1, \sigma_1)(x)$ şi $N_2(\mu_2, \sigma_2)(x)$, iar ecuaţia (8.16) devine:

$$P_1 \frac{1}{\sigma_1\sqrt{2\pi}} e^{-\frac{(T-\mu_1)^2}{2\sigma_1^2}} = P_2 \frac{1}{\sigma_2\sqrt{2\pi}} e^{-\frac{(T-\mu_2)^2}{2\sigma_2^2}}$$

Prin logaritmare, se obţine urmatoarea ecuaţie de gradul 2 în necunoscuta T:

$$T^2\left(\frac{1}{\sigma_1^2} - \frac{1}{\sigma_2^2}\right) - 2T\left(\frac{\mu_1}{\sigma_1^2} - \frac{\mu_2}{\sigma_2^2}\right) + \left(\frac{\mu_1^2}{\sigma_1^2} - \frac{\mu_2^2}{\sigma_2^2}\right) - 2\ln\frac{P_1}{P_2}\frac{\sigma_2}{\sigma_1} = 0$$

Una dintre simplificările uzuale este presupunerea că $\sigma_1 = \sigma_2 = \sigma$; această presupunere implică modelarea imaginii în nivele de gri ca o imagine cu doar două nivele de gri μ_1 şi μ_2, afectată de un zgomot Gaussian aditiv, având varianţa σ^2. În aceste condiţii, ecuaţia de gradul 2 devine o ecuaţie liniară, a carei soluţie este:

$$T = \frac{\mu_1 + \mu_2}{2} - \frac{\sigma^2}{\mu_1 - \mu_2}\ln\frac{P_1}{P_2}$$

Metoda se poate extinde şi pentru imagini ce conţin mai mult de două tipuri de obiecte; în acest caz este însă necesară presupunerea suplimentară de localizare a modurilor, astfel încât să se poată considera, ca şi în cazul metodei Bhattacharyya, că influenţa fiecărui mod este limitată la intervale nesuprapuse de nivele de gri.

8.1.2 Creşterea şi fuziunea regiunilor

Pentru aplicarea cu succes a tehnicilor de segmentare pe histogramă prezentate anterior trebuiesc îndeplinite neapărat câteva condiţii (deja enunţate). Aplicarea tehnicilor de segmentare pe histogramă este condiţionată în primul rând de reprezentarea diferitelor clase de obiecte din imagine pe intervale de nivele de gri diferite care nu se suprapun (sau se suprapun parţial pe porţiuni foarte mici); apoi este necesară cunoaşterea numărului de tipuri de obiecte diferite. În fine, se presupune că valorile prag corespunzătoare se pot determina cu o precizie corespunzătoare.

Chiar în cazurile în care toate aceste condiţii enunţate sunt îndeplinite, nu se poate garanta condiţia de conexitate a regiunilor obţinute în urma segmentării (8.2). Acest lucru este evident, atât timp cât două obiecte de acelaşi tip, neconexe, primesc prin segmentarea pe histogramă o aceeaşi etichetă, şi formează în imaginea de etichete o regiune neconexă. O metodă care respectă toate condiţiile impuse de definiţia metematică a segmentării, şi anume (8.1), (8.2) şi (8.3), este creşterea regiunilor.

Creşterea regiunilor

Principiul pe care se bazează creşterea regiunilor este simplu: se aleg în imagine puncte reprezentative pentru fiecare obiect individual şi categorie de obiecte, pe baza cărora are loc un proces de aglomerare a pixelilor vecini acestora, ce au aceleaşi proprietăţi (în particular acelaşi nivel de gri). În urma acestui proces de aglomerare (adăugare de puncte) se obţin zone (regiuni) de pixeli cu aceleaşi caracteristici, deci obiecte individuale. Procesul se opreşte în momentul în care fiecare punct al imaginii a fost alocat unei regiuni. Evident, metoda astfel descrisă pe scurt, are două etape esenţiale: alegerea punctelor de start (puncte iniţiale), numite germeni sau seminţe, şi creşterea propriu-zisă a regiunilor [19], [2].

Numărul final de regiuni rezultate este egal cu numărul de germeni aleşi iniţial pentru creştere. În principiu, este de dorit ca fiecare obiect individual aflat în imagine să fie marcat de câte un germene. Dacă în interiorul unui aceluiaşi obiect se găsesc mai mulţi germeni, pentru fiecare dintre ei va fi crescută o regiune; acesta face ca obiectul iniţial să fie împărţit artificial prin segmentare în mai multe regiuni. Parţial, acest neajuns se poate corecta printr-o etapă ce urmează creşterii regiunilor, şi anume fuziunea regiunilor adiacente ce au proprietăţi asemănătoare. Dacă în interiorul unui obiect nu este ales nici un germene, obiectul respectiv va fi înglobat de regiunile ce cresc pornind de la germeni din vecinătatea sa spaţială; astfel, respectivul obiect nu apare ca o regiune distinctă şi este pierdut, rezultând o eroare gravă de segmentare.

Pentru a preveni efectul unor neuniformităţi de iluminare pe suprafaţa imaginii, acesta este împărţită în ferestre nesuprapuse; în fiecare astfel de fereastră se alege un număr de germeni, al căror plasament spaţial este aleator (germenii se distribuie uniform pe

suprafaţa imaginii). Germenii se aleg astfel încât nivelul lor de gri să fie reprezentativ pentru obiectele prezente local (deci nivelul de gri al germenilor trebuie să corespundă unor maxime ale histogramei locale). În plus, trebuie verificat ca plasamentul spaţial al germenilor să se facă în interiorul regiunilor şi nu pe frontiera acestora. Verificarea se poate face simplu pe baza calculului unui operator derivativ local, ca de exemplu laplacianul (37); dacă valoarea acestuia nu depăşeşte un anumit procent prestabilit (10% - 20%) din diferenţa maximă de nivele de gri a ferestrei, punctul ales este considerat ca plasat corect.

O verificare suplimentară încearcă să prevină o eventuală suprasegmentare[2] (împărţirea artificială a unui acelaşi obiect în mai multe regiuni), eliminând germenii plasaţi în interiorul aceluiaşi obiect. Verificarea se face pe baza calculului variaţiei nivelelor de gri de-a lungul drumurilor[3] arbitrare ce unesc perechi de germeni. Dacă există o cale ce uneşte doi germeni de-a lungul căreia nivelul de gri nu variază cu mai mult de 20% - 30% din diferenţa maximă a nivelelor de gri din fereastră, cei doi germeni sunt plasaţi în interiorul unei zone de nivele de gri uniforme, deci în interiorul unui acelaşi obiect. În aceste condiţii unul dintre cei doi germeni ai perechii este eliminat, deoarece este redundant. Dacă de-a lungul tuturor căilor ce unesc perechea de germeni nivelul de gri variază mai mult decât pragul ales, atunci se consideră că cei doi germeni sunt plasaţi în interiorul unor obiecte diferite (deoarece căile ce unesc germenii traversează regiuni de frontieră). În practică, examinarea tuturor drumurilor (căilor) ce unesc perechi de germeni este extrem de costisitoare din punctul de vedere al timpului de calcul. De aceea se verifică doar căile formate din segmente verticale şi orizontale, şi eventual, dreapta ce uneşte cele două puncte (dacă această dreaptă poate fi reprezentată de o secvenţă de puncte conexe) (vezi figura 8.5).

Valorile procentuale ale pragurilor de comparaţie, precum şi numărul de germeni distincţi ce rămân după procesul de reducere, nu trebuie considerate ca fixe; nu există valori standardizate şi alegerea acestora se face pe baza condiţiilor particulare (legate de conţinutul imaginii) şi a experienţei utilizatorului.

Pornind de la germenii aleşi, regiunile sunt obţinute printr-un proces de creştere aproape simultană, început de la aceştia, până când toţi pixelii imaginii sunt repartizaţi unei regiuni. Cvasi-simultaneitatea creşterii poate fi realizată cu un algoritm serial, prin alocarea pixelilor ce sunt adiacenţi (vecini) zonelor deja segmentate. Acestă alocare trebuie să ţină seama de criteriul ca regiunile crescute să fie uniforme: nivelul de gri al pixelului ce se adaugă nu trebuie să difere cu mai mult de un prag prestabilit faţă de nivelul de gri al germenului regiunii la care se alocă. În acelaşi timp, la o singură trecere, numărul de puncte ce se adaugă unei regiuni nu poate depăşi un număr prestabilit (condiţia încearcă

[2] În general, prin suprasegmentare se înţelege partiţionarea imaginii într-un număr de regiuni mai mare decât numărul de obiecte. Există şi noţiunea reciprocă de subsegmentare: împărţirea imaginii într-un număr de regiuni mai mic ca numărul de obiecte.

[3] Un drum între două puncte ale imaginii este o secvenţă ordonată de puncte ale imaginii, vecine două câte două (relativ la un anumit tip de conexitate, V_4 sau V_8), care are drept capete punctele considerate.

Fig. 8.5: Reducerea numărului de germeni: germenii 1 şi 2 sunt uniţi de o cale cu segmente paralele cu orizontala şi verticala de intensitate constantă, deci sunt redundanţi; germenii 3 şi 4 sunt uniţi de o cale dreaptă de aceeşi intensitate, deci sunt redundanţi; orice cale ce uneşte germenii 1 şi 3 are o diferenţă mare de intensitate.

să asigure creşterea relativ uniformă şi izotropă a tuturor regiunilor).

Dacă adăugarea de noi pixeli se blochează (criteriul de uniformitate nu mai este respectat), diferenţa maxim admisă pentru nivelul de gri poate fi crescută în etape, până la epuizarea pixelilor imaginii.

Avantajele pe care le are o asemenea tehnică de creştere a regiunilor sunt acelea că nu mai este necesară nici o informaţie privind conţinutul imaginii, regiunile crescute sunt conexe şi nu există puncte neetichetate (nealocate vreunei regiuni) şi poziţia frontierelor dintre diferitele regiuni corespunde poziţiei frontierelor percepute subiectiv în imagine.

Fuziunea regiunilor

O extindere a principiului utilizat în creşterea regiunilor, şi anume adăugarea la o regiune a unor entităţi (pixeli în acest caz) a căror proprietăţi sunt similare cu cele ale unui obiectului de bază (regiunea), se află la baza tehnicilor de fuziune a regiunilor [9]. Fuziunea regiunilor constă în reunirea iterativă a regiunilor adiacente (începând de la nivelul unor entităţi atomice ale imaginii - deci pixelii) până când regiunile adiacente devin suficient de diferite. Procesul de fuziune a regiunilor poate fi aplicat şi în urma unei creşteri a regiunilor, pentru a înlătura efectele unei eventuale suprasegmentări. Există mai multe criterii de fuziune a regiunilor adiacente, a căror acţiune de verificare a deosebirii între regiuni se face fie prin inspecţia frontierei comune, fie prin caracterizarea interiorului regiunii.

Pentru două regiuni adiacente R_i şi R_j, al căror perimetru este $Perim(R_i)$ şi $Perim(R_j)$, putem determina $P_m = \min(Perim(R_i), Perim(R_j))$ şi P lungimea frontierei comune[4].

[4]Lungimea frontierei comune poate fi măsurată fie ca perimetru, fie ca număr de puncte ce o compun.

Pe această frontieră comună se disting puncte *slabe* (în număr de n_s) şi puncte *tari* (în număr de n_t). Un punct slab este acel punct pentru care diferenţa nivelelor de gri între vecinii din regiunile adiacente este foarte mică (mai mică decât un anumit prag fixat). Un punct tare este acel punct pentru care diferenţa de nivele de gri între vecinii din regiunile adiacente este foarte mare (mai mare ca un anumit prag fixat). Cu aceste notaţii, criteriile de fuziune a regiunilor R_i şi R_j sunt:

- dacă numărul de puncte slabe raportat la perimetrul minim este important, $\frac{n_s}{P_m} > \theta_1$
- dacă numărul de puncte slabe de pe frontiera comună este mare, $\frac{n_s}{P} > \theta_2$
- dacă numărul de puncte tari de pe frontiera comună este mic, $\frac{n_t}{P} < \theta_3$.

Parametrul θ_1 controlează dimensiunea regiunilor ce se unesc şi se alege în general cu valoarea 0.5 (de exemplu o valoare apropiată de 1 implică unirea a două regiuni numai dacă una dintre ele este aproape înconjurată de cealaltă). Valori tipice pentru parametrii θ_2 şi θ_3 sunt 0.75 şi 0.2.

Abordarea fuziunii pe baza caracterizării interiorului regiunilor necesită definirea a două componente: o modalitate de caracterizare a proprietăţilor regiunilor şi o modalitate de a defini "apropierea" sau similaritatea dintre trăsături în termeni numerici.

Vectorul de trăsături ce caracterizează o regiune se compune din momente statistice ale variabilei aleatoare ale cărei realizări particulare sunt nivelele de gri din regiune repectivă; nu pot lipsi din acest vector nivelul de gri mediu al regiunii şi varianţa acestuia.

În [9] se propun patru funcţii de măsură a asemănării între perechi de vectori; pentru doi vectori $\mathbf{x}_i$ şi $\mathbf{x}_j$, având aceeaşi dimensiune, acestea se definesc ca:

$$F_1(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle \text{ (produsul scalar dintre vectori)}$$

$$F_2(\mathbf{x}_i, \mathbf{x}_j) = \frac{\langle \mathbf{x}_i, \mathbf{x}_j \rangle}{\langle \mathbf{x}_i, \mathbf{x}_i \rangle + \langle \mathbf{x}_j, \mathbf{x}_j \rangle - \langle \mathbf{x}_i, \mathbf{x}_j \rangle} \text{ (similaritatea dintre vectori)}$$

$$F_3(\mathbf{x}_i, \mathbf{x}_j) = \frac{\langle \mathbf{x}_i, \mathbf{x}_j \rangle}{\sqrt{\langle \mathbf{x}_i, \mathbf{x}_i \rangle \langle \mathbf{x}_j, \mathbf{x}_j \rangle}} \text{ (corelaţia normalizată dintre vectori)}$$

$$F_4(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j) \cdot A \cdot (\mathbf{x}_i - \mathbf{x}_j)^T$$

(distanţa generalizată dintre vectori, unde A este o matrice pozitiv definită)

Pentru primele trei funcţii, o valoare mai mare corespunde unei asemănări mai mari între vectori (valorile maxime pentru $F_2(\mathbf{x}_i, \mathbf{x}_j)$ şi $F_3(\mathbf{x}_i, \mathbf{x}_j)$ sunt 1). Pentru funcţia de similaritate bazată pe distanţa generalizată, o valoare mai mică corepunde unei asemănări mai puternice între vectori. Prin particularizarea matricii A se pot obţine diferite distanţe, ca distanţa Euclidiană obişnuită (A fiind matricea unitate, $A = I$), distanţe Euclidiene ponderate (dacă A este o matrice diagonală), sau distanţa Mahalanobis (dacă A este o matrice de covariaţie a componentelor).

8.2 Segmentarea orientată pe contururi

Într-o imagine, variaţiile de valoare ale pixelilor reprezintă schimbări ale proprietăţilor fizice sau geometrice ale scenei sau ale obiectului observat. Aceste schimbări pot corespunde fizic la variaţiile iluminării, schimbările de orientare sau de distanţă faţă de observator, schimbări de reflectanţă ale suprafeţelor, variaţii de absorbţie a radiaţiei. Într-un număr mare de cazuri, aceste variaţii de intensitate sunt informaţii importante pentru operaţiile ce urmează segmentării, informaţii ce corespund frontierelor regiunilor determinate de obiectele scenei.

8.2.1 Metode derivative

Principiul acestei metode constă în definirea punctelor de contur ca fiind acei pixeli ai imaginii în care apar schimbări importante (abrupte) ale nivelului de gri. Deci, măsurarea acestei variaţii se va face prin operatori derivativi de tip gradient.

Pentru o imagine cu suport spaţial continuu, pe direcţia unei muchii, derivata va fi maximă. Derivata imaginii pe direcţia r, ce face unghiul θ cu orizontala, este dată de combinaţia liniară a derivatelor parţiale pe direcţiile orizontală şi verticală (8.17):

$$\frac{\partial f}{\partial r} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial r} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial r} = \frac{\partial f}{\partial x}\cos\theta + \frac{\partial f}{\partial y}\sin\theta$$

$$\frac{\partial f}{\partial r} = f_x \cos\theta + f_y \sin\theta \tag{8.17}$$

Valoarea maximă a acestei derivate, calculate după unghiul θ este determinată de ecuaţia

$$\frac{\partial}{\partial \theta}\left(\frac{\partial f}{\partial r}\right) = -f_x \sin\theta + f_y \cos\theta = 0$$

ce are soluţia evidentă:

$$\theta_0 = \arctan\left(\frac{f_y}{f_x}\right) \tag{8.18}$$

Pe această direcţie, modulul gradientului este:

$$\left(\frac{\partial f}{\partial r}\right)_{\max} = \sqrt{f_x^2 + f_y^2} \tag{8.19}$$

Din punct de vedere practic, implementarea acestei metode implică atunci calcularea, pentru fiecare punct al imaginii, a derivatelor parţiale f_x şi f_y, calcularea modulului gradientului maxim (8.19) şi a direcţiei acestuia (8.18). Valoarea gradientului maxim din fiecare punct al imaginii este apoi comparată cu un prag fixat: dacă pragul este depăşit

(deci gradientul maxim în pixelul respectiv este suficient de important) atunci pixelul testat este pixel de contur.

Realizarea derivatelor parţiale după direcţiile orizontală şi verticală implică translaţia în discret a lui f_x şi f_y:

$$f_x = \frac{\partial f}{\partial x} = \frac{\Delta f(m,n)}{\Delta m}$$
$$f_y = \frac{\partial f}{\partial y} = \frac{\Delta f(m,n)}{\Delta n}$$

Aceste derivate parţiale discrete pot avea mai multe implementări:

$$f_x = f(m,n) - f(m+1,n),\ f_y = f(m,n) - f(m,n+1) \tag{8.20}$$

$$f_x = f(m-1,n) - f(m,n),\ f_y = f(m,n-1) - f(m,n) \tag{8.21}$$

$$f_x = f(m-1,n) - f(m+1,n),\ f_y = f(m,n-1) - f(m,n+1) \tag{8.22}$$

Toate expresiile date de (8.20), (8.21), (8.22) sunt combinaţii liniare ale valorilor unor pixeli din imagine, situaţi în vecinătatea pixelului curent din poziţia (m,n). Deci toate aceste operaţii se pot realiza prin filtrări liniare cu măşti potrivite: (8.23) pentru (8.20), (8.24) pentru (8.21), (8.25) pentru (8.22).

$$W_x = \begin{pmatrix} 1 & -1 \end{pmatrix},\ W_y = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \tag{8.23}$$

$$W_x = \begin{pmatrix} 1 & -1 \end{pmatrix},\ W_y = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \tag{8.24}$$

$$W_x = \begin{pmatrix} 1 & 0 & -1 \end{pmatrix},\ W_y = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} \tag{8.25}$$

Schema bloc a extragerii de contururi este reprezentată în figura 8.6.

Harta de orientări este o imagine care conţine, pentru fiecare pixel, orientarea gradientului de modul maxim în punctul respectiv, şi este în general folosită la prelucrarea suplimentară a contururilor (conectare de contururi, extragere direcţională de contururi). Harta de contururi este o imagine binară în care punctele marcate (puncte-obiect) corespund poziţiei punctelor de contur (puncte cu gradient de modul mare). O simplificare uzuală practicată este înlocuirea normei L2 din calculul modulului maxim al gradientului (8.19) cu norma L1, ceea ce conduce la aproximarea:

$$\left(\frac{\partial f}{\partial r}\right)_{\max} \approx |f_x| + |f_y|$$

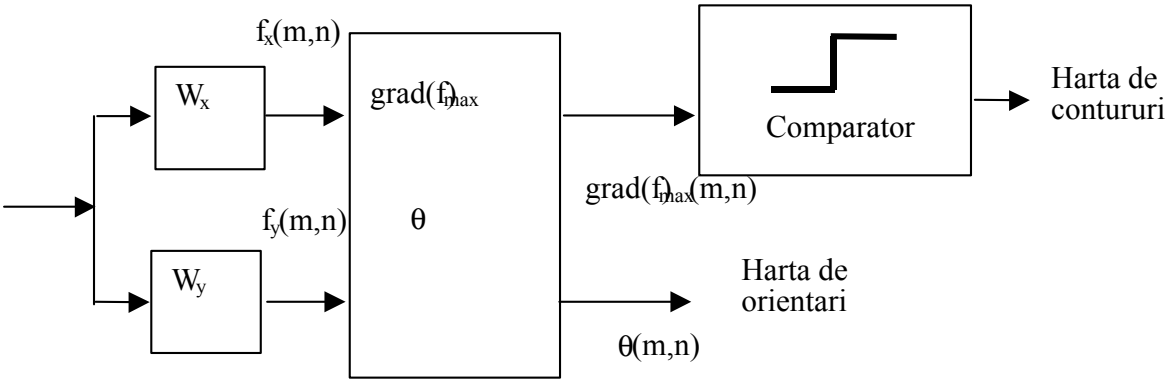

Fig. 8.6: Schema bloc a extractorului de contururi bazat pe metoda de gradient.

Folosirea măştilor de derivare pe verticală şi orizontală prezentate are însă serioase neajunsuri: dimensiunea lor mică face ca rezultatele să fie extrem de sensibile în prezenţa zgomotului. În aceste condiţii a apărut naturală ideea de a combina filtrarea de derivare cu o filtrare de netezire, care să mai reducă efectele zgomotului. Considerând zgomotul de tip gaussian, aditiv, filtrarea de netezire are ca efect secundar micşorarea contrastului frontierelor obiectelor din imagine (efectul de înceţoşare, sau *blur*). Pentru ca în aceste condiţii detecţia contururilor să nu fie afectată, trebuie ca operaţia de mediere prin care se realizează netezirea să se facă pe o direcţie perpendiculară direcţiei contururilor căutate [3]. Atunci derivarea pe verticală se combină cu o operaţie de netezire cu mască orizontală $\begin{pmatrix} 1/3 & \boxed{1/3} & 1/3 \end{pmatrix}$ şi derivarea pe orizontală se combină cu o operaţie de netezire cu mască verticală $\begin{pmatrix} 1/3 \\ \boxed{1/3} \\ 1/3 \end{pmatrix}$. Dacă folosim pentru derivare masca W_y din (8.23), masca de filtrare rezultantă va fi $\begin{pmatrix} 1/3 & \boxed{1/3} & 1/3 \\ -1/3 & -1/3 & -1/3 \end{pmatrix}$. În cazul general se pot folosi însă pentru netezire medieri ponderate (şi nu neapărat medieri aritmetice), care să acorde o mai mare importanţă pixelului curent prelucrat, ca de exemplu $\frac{1}{c+2}\begin{pmatrix} 1 & \boxed{c} & 1 \end{pmatrix}$ şi se preferă folosirea operatorilor de derivare simetrici, de tipul (8.25). Ceea ce rezultă pentru operatorii de derivare orizontală şi verticală sunt măştile:

$$W_x = \begin{pmatrix} 1 & 0 & -1 \\ c & \boxed{0} & -c \\ 1 & 0 & -1 \end{pmatrix}, \; W_y = \begin{pmatrix} 1 & c & 1 \\ 0 & \boxed{0} & 0 \\ -1 & -c & -1 \end{pmatrix} \tag{8.26}$$

Prin particularizarea valorilor constantei de ponderare c se pot obţine diferite tipuri de operatori de extragere de contur clasici: Prewitt ($c = 1$), Izotrop ($c = \sqrt{2}$), Sobel ($c = 2$) [9]. Se remarcă faptul că constanta de ponderare globală a măştii de filtare este neesenţială, întrucât condiţia de normare ce trebuie îndeplinită este cea pentru filtre de contrastare (derivare) (3.9): suma coeficienţilor măştii să fie nulă. Figura 8.7 prezintă

harta de intensitate a tranziţiilor (modulul maxim al gradientului, $\left(\frac{\partial f}{\partial r}\right)_{\max}$) iar figura 8.8 prezintă harta binară de contururi extrasă prin compararea hărţii de intensităţii cu un prag fixat (binarizarea hărţii de intensitate).

Fig. 8.7: Harta de intensitate a contururilor (modulul maxim al gradientului) calculat cu măşti Prewitt pentru imaginea "lena".

Fig. 8.8: Harta binară de contururi extrasă din harta de intensităţi precedentă.

Informaţia de orientare este în general folosită în etape următoare ale prelucrării; unghiurile determinate după (8.18) oferă un unghi "exact" al direcţiei conturului în punctul curent, calculat cu un efort semnificativ de calcul (împărţire şi calcul de arctangentă). În practică, această informaţie este prea exactă: pe grila pătrată de eşantionare nu se pot reprezenta cu uşurinţă drepte continue după orice direcţie[5]; câteva direcţii sunt fa-

[5]Problema trasării figurilor geometrice oarecari (inclusiv a dreptelor) este rezolvată de grafica pe

vorizate şi uşor de utilizat (vertical, orizontal, cele două diagonale). În acest caz se poate măsura în fiecare modulul gradientului după aceste câteva direcţii importante, şi apoi se poate alege direcţia după care acest modul este maxim. Acesta este principul operatorilor compas.

Un operator compas este definit de un număr de măşti de derivare (corespunzătoare în continuare unor filtrări liniare) pe direcţiile principale (vertical, orizontal, cele două diagonale), în cele două sensuri. Compasul clasic are $D = 8$ măşti de filtrare (identice două câte două, mai puţin semnul), fiecare dintre ele realizând o derivare după o direcţie multiplu de 45°. Schema bloc a unui operator compas este prezentată în figura 8.9; se remarcă faptul că, odată determinată valoarea maximă a modulului gradientului în pixelul curent (m, n), obţinerea hărţii de contururi se face ca şi la un operator de gradient clasic.

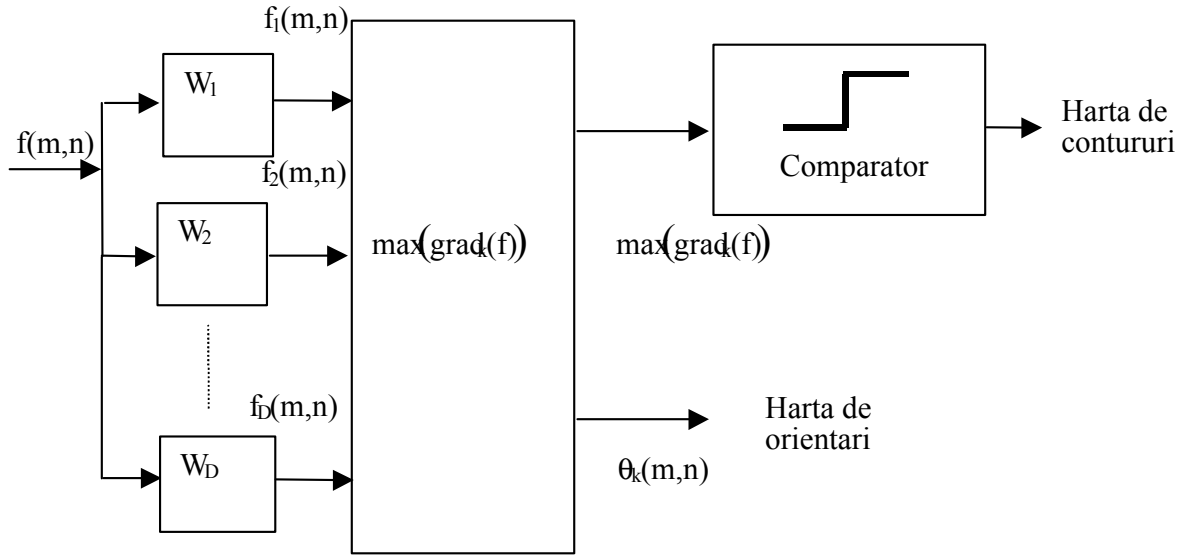

Fig. 8.9: Schema bloc a unui operator compas de extragere a contururilor.

Un exemplu de măşti de derivare direcţională sunt măştile următoare (indexate după direcţia geografică pe care calculează derivata): $W_N = \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}$, $W_{NV} = \begin{pmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$, $W_V = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix}$, $W_{SV} = \begin{pmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{pmatrix}$, $W_S = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}$, $W_{SE} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{pmatrix}$, $W_E = \begin{pmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{pmatrix}$, $W_{NE} = \begin{pmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{pmatrix}$. După cum

calculator, prin algoritmi de rendering.

se remarcă, familia de măşti se poate genera pornind de la una dintre măştile Prewitt, prin translaţii circulare cu o poziţie a frontierei măştii în jurul centrului ei; în mod analog se pot obţine operatori compas bazaţi pe masca Sobel sau pe gradientul izotrop sau pe masca Kirsch $\begin{pmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{pmatrix}$. Precizia unghiulară a operatorilor compas este deci determinată de numărul de orientări diferite pe care se calculează derivatele, şi deci de numărul de translaţii ale frontierei măştii; pentru o mască pătrată de bază de dimensiune N, precizia unghiulară a operatorului compas este de $90°/(N-1)$.

Unul dintre principalele dezavantaje ale metodelor de gradient este precizia slabă de localizare a conturului (a centrului tranziţiei) în condiţiile unei pante puţin abrupte a acestuia (tranziţii slabe, graduale). Derivata a doua poate fi însă folosită pentru a determina capetele tranziţiei (cele două extreme), sau pentru a marca centrul tranziţiei (trecerea sa prin zero); figura 8.10 ilustrează această comportare pentru cazul unidimensional.

Operatorul bazat pe trecerea prin zero a derivatei secunde este operatorul "zero-crossing" [9]. În cazul imaginilor (semnale cu suport bidimensional) trebuie luată în considerare derivata secundă după ambele direcţii, combinate în laplacian:

$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

În cazul discret, măşti ce implementează laplacianul sunt măştile $W_5 - W_7$, prezentate la capitolul de îmbunătăţire a contrastului imaginilor (pag. 37). Precizia sporită a operatorilor laplacieni conduce însă la o sensibilitate crescută în prezenţa zgomotelor (mai mare decât a operatorilor de gradient). Mai mult, laplacianul nu mai conţine informaţie relativă la direcţia tranziţiei.

8.2.2 Alte metode

O clasă importantă de operaţii neliniare de extragere a contururilor sunt cele bazate pe morfologia matematică. În secţiunea 6.2.3 am prezentat operatori morfologici de extragere a contururilor. Principiul acestora este de a măsura diferenţele dintre valorile extreme (minim şi maxim) ale vecinătăţii punctului curent; dacă diferenţa dintre aceste valori este suficient de mare înseamnă că punctul curent este un punct de contur, aflându-se într-o zonă de tranziţie a valorilor pixelilor. Variante ale acestei tehnici de bază se pot obţine prin considerarea a mai multe elemente structurante, având diferite forme şi dimensiuni.

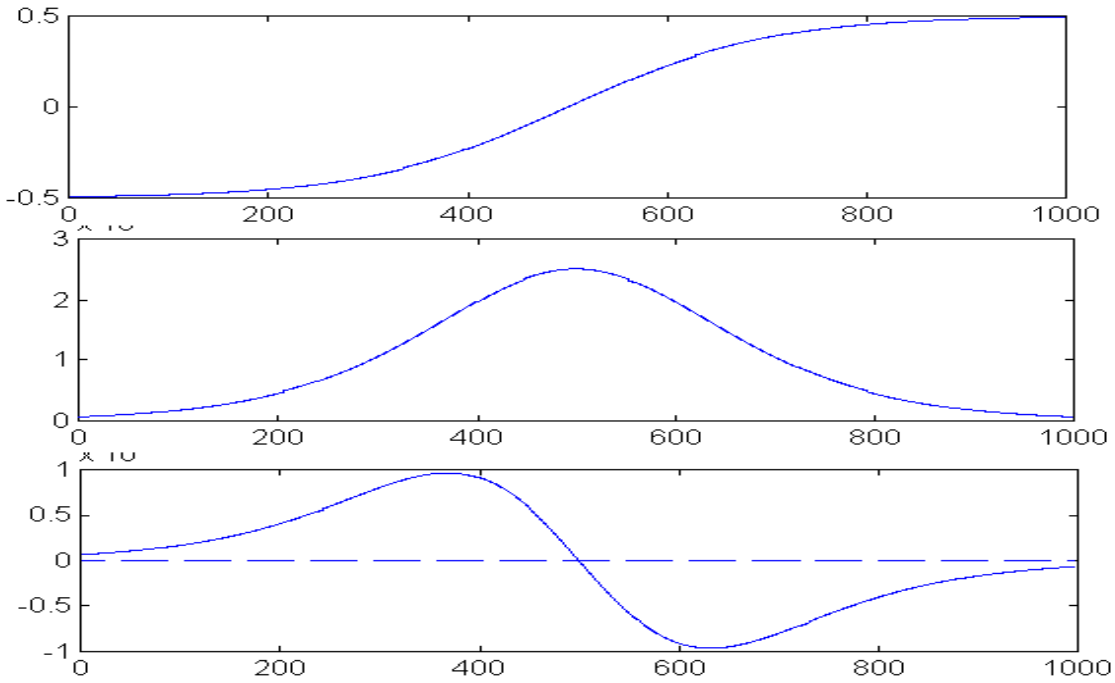

Fig. 8.10: Profil de tranzitie graduală; maximul primei derivate nu poate marca cu precizie centrul tranziţiei; derivata secundă trece prin zero la mijlocul conturului.

Capitolul 9

PARAMETRI DE FORMĂ

Prin parametri de formă înţelegem în general orice scalar sau funcţie (cu suport unidimensional sau bidimensional) asociate unei forme plane pe care o caracterizează; forme asemănătoare sunt caracterizate de parametri de formă de valori apropiate; formele diferite prezintă diferenţe mari între parametrii de formă ce le sunt asociaţi. Parametrii de formă compun un fel de fişă de identitate a formei respective, pe baza cărei această formă poate fi recunoscută în mod unic. În mod ideal, aceşti parametri trebuie să fie invarianţi la translaţie, rotaţie şi scalare. Tehnicile de recunoaştere a formelor sau de clasificare sunt precedate întotdeauna de o etapă de extragere a parametrilor de formă (sau a caracteristicilor formei).

Pentru analiza imaginilor, o formă este o funcţie de două variabile, cu suport compact $f(x, y) : K \rightarrow \mathbf{R}$; în general valorile acestei funcţii sunt binare (0 sau 1), descriind deci o parte a unei imagini binare (zona din imagine în care se află obiectul de interes). Atunci funcţia poate fi văzută ca o funcţie caracteristică a formei, asemănătoare funcţiei caracteristice a unei mulţimi.

În cele ce urmează vom prezenta câţiva parametri de formă clasici.

9.1 Parametri geometrici

Această categorie de parametri se bazează pe măsura unor atribute geometrice simple: arie (S), perimetru (P), număr de găuri, numărul lui Euler (numărul de regiuni conexe – numărul de găuri). Cum nu toate aceste numere sunt invariante şi caracteristice unic unei anume forme, au apărut combinaţii de tip raport.

Raportul de compacitate (numit şi factor de formă [19]) este raportul dintre pătratul

perimetrului şi suprafaţa formei:

$$\kappa = \frac{P^2}{4\pi S} \tag{9.1}$$

Pentru o formă circulară raportul este unitar; cu cât numărul κ este mai apropiat de această valoare, cu atât mai mult forma seamănă cu un disc (pătratul are un raport de compacitate $\kappa = 1.273$). Există însă forme diferite caracterizate de aceeaşi valoarea a parametrului dat de (9.1).

Excentricitatea sau circularitatea formei (măsura în care forma dată se deosebeşte de disc) poate fi definită şi ca un raport al razelor cercurilor circumscrise (R) şi înscrise (r) formei:

$$c = \frac{R}{r} \tag{9.2}$$

Acest raport este evident unitar în cazul discului; pentru pătrat valoarea sa este de $c = 1.412$.

9.2 Momente statistice şi invarianţi

Interpretând funcţia caracteristică a formei ca pe o funcţie de densitate de probabilitate bidimensională, putem defini momentele statistice asociate celor două variabile aleatoare (ce sunt coordonatele punctelor formei) [17]:

$$m_{pq} = \iint\limits_K f(x,y) x^p y^q dx dy,\ p,q = 0,1,2,... \tag{9.3}$$

Scalarul m_{pq} (momentul de ordin p,q sau $p+q$) este proiecţia funcţiei $f(x,y)$ pe polinoamele x^p şi y^q ale bazei complete de polinoame. Teorema reprezentării cu momente afirmă că mulţimea infinită de momente m_{pq} determină în mod unic $f(x,y)$ şi reciproc.

În cazul imaginilor binare, coordonatele sunt discrete şi funcţia este o funcţie caracteristică; formula momentelor (9.3) devine

$$m_{pq} = \sum_{f(x,y)\neq 0} \sum x^p y^q \tag{9.4}$$

Cum caracterizarea unei forme printr-o serie infinită de numere (aşa cum cere teorema reprezentării cu momente) nu este posibilă, în practică se folosesc serii de momente truncheate până la un ordin maxim fixat N ($p+q \leqslant N$). Acestea însă caracterizează o altă funcţie, $g(x,y)$, o aproximare a lui $f(x,y)$. Această aproximare este dată de o combinaţie liniară a polinoamelor bazei, ponderate cu scalarii necunoascuţi g_{pq}:

$$g(x,y) = \sum_{p+q\leqslant N} \sum g_{pq} x^p y^q \tag{9.5}$$

Găsirea acestor scalari se face prin egalarea momentelor cunoscute ale lui $f(x,y)$ cu momentele lui $g(x,y)$ dată de expresia (9.5). Rezolvând sistemul de ecuaţii cuplate ce se formează (a se vedea [9]), se pot obţine relaţiile căutate; calculul trebuie însă refăcut, din cauza cuplării ecuaţiilor, ori de câte ori se doreşte trecerea la o aproximare mai bună a formei f , mărind valoarea lui N. Această cuplare provine din cauza folosirii unei baze neortogonale (aşa cum este familia de polinoame $x^p y^q$) pentru calculul momentelor; problema a fost rezolvată prin folosirea proiecţiilor pe baza de polinoame Legendre [9].

Trebuie însă remarcat că folosirea momentelor statistice pentru caracterizarea unei forme nu asigură îndeplinirea a nici unuia dintre principiile de invarianţă căutate; de aceea au fost introduse momente statistice invariante [19], [9].

Momentele statistice invariante la translaţie sunt momentele statistice centrate:

$$\mu_{pq} = \sum_{f(x,y)\neq 0} \sum (x-\overline{x})^p (y-\overline{y})^q \tag{9.6}$$

Momentele statistice invariante la translaţie şi scalare sunt definite de:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}},\ \gamma = 1 + \frac{p+q}{2} \tag{9.7}$$

Invarianţii la translaţie, scalare şi rotaţie ai unei forme, obţinuţi în condiţiile folosirii unor momente statistice de ordin cel mult 3 ($N = 3$), sunt în număr de 7 şi sunt exprimaţi de:

$$\Phi_1 = \eta_{20} + \eta_{02} \tag{9.8}$$

$$\Phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \tag{9.9}$$

$$\Phi_3 = (\eta_{30} - 3\eta_{12})^2 + (\eta_{03} - 3\eta_{21})^2 \tag{9.10}$$

$$\Phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{03} + \eta_{21})^2 \tag{9.11}$$

$$\begin{aligned}\Phi_5 = {} & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12}) \left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{03} + \eta_{21})^2\right] + \\ & + (\eta_{03} - 3\eta_{21})(\eta_{03} + \eta_{21}) \left[(\eta_{03} + \eta_{21})^2 - 3(\eta_{30} + \eta_{12})^2\right]\end{aligned} \tag{9.12}$$

$$\Phi_6 = (\eta_{20} - \eta_{02}) \left[(\eta_{30} + \eta_{12})^2 - (\eta_{03} + \eta_{21})^2\right] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{03} + \eta_{21}) \tag{9.13}$$

$$\begin{aligned}\Phi_7 = {} & (\eta_{30} - 3\eta_{21})(\eta_{03} + \eta_{21}) \left[(\eta_{03} + \eta_{21})^2 - 3(\eta_{30} + \eta_{12})^2\right] - \\ & - (\eta_{03} - 3\eta_{21})(\eta_{30} + \eta_{12}) \left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{03} + \eta_{21})^2\right]\end{aligned} \tag{9.14}$$

Iniţial (mijlocul anilor '60) aceşti invarianţi au fost folosiţi pentru recunoaşterea caracterelor mari de tipar, cu rezultate modeste. Eficienţa lor constă însă în modul rapid de calcul şi posibilitatea de a le utiliza cu succes pentru recunoaşterea formelor geometrice convexe.

Folosind momentele invariante, se mai pot deduce alte atribute: excentricitatea suprafeţei (9.15), care măsoară gradul de uniformitate al distribuţiei punctelor formei în jurul centrului de greutate şi orientarea suprafeţei, caracterizată de unghiul θ faţă de orizontală al axei faţă de care momentul inerţie al formei este minim (9.16).

$$\varepsilon = \frac{\Phi_2}{\mu_{00}} \tag{9.15}$$

$$\theta = \frac{1}{2} \arctan \frac{2\mu_{11}}{\mu_{20} - \mu_{02}} \tag{9.16}$$

9.3 Semnătura formei

Semnătura unei forme este o funcţie scalară de o variabilă, asociată unei forma plane. Semnătura este definită de distanţa de la un punct de referinţă fixat $\mathbf{x}$ (în general centrul de greutate al formei) la fiecare punct de pe conturul (frontiera) formei. Această distanţă este exprimată (sau măsurată) în funcţie de unghiul la centru θ realizat de punctul curent de pe contur cu axa orizontală de referinţă, $d_{\mathbf{x}}(\theta)$ sau în funcţie de abscisa curbilinie ρ (lungimea conturului cuprins între punctul curent şi punctul în care axa de referinţă intersectează conturul), $d_{\mathbf{x}}(\rho)$. Semnătura formei este o reprezentare reversibilă (cunoscând semnătura se poate reconstrui conturul obiectului). Figurile 9.1 şi 9.2 prezintă semnăturile unor forme poligonale.

Fig. 9.1: Semnătura unui pătrat ca funcţie de unghiul la centru; punctul de referinţă este centrul de greutate, axa de referinţă este orizontala.

Este posibil ca, pentru formele concave, semnătura în funcţie de unghiul la centru să nu poată fi calculată, deoarce, pentru anumite unghiuri θ, intersecţia dreptei de direcţie θ cu conturul să fie formată din mai multe puncte. Această problemă nu apare în cazul semnăturii în funcţie de abscisa curbilinie. Ca o restricţie generală, în cazul folosirii semnăturii bazate pe unghi, trebuie ca punctul de referinţă $\mathbf{x}$ să aparţină nucleului formei. Nucleul formei K, $Ker(K)$ este definit prin:

$$Ker(K) = \{\mathbf{x} | \forall \mathbf{y} \in K, [\mathbf{xy}] \subseteq K\} \tag{9.17}$$

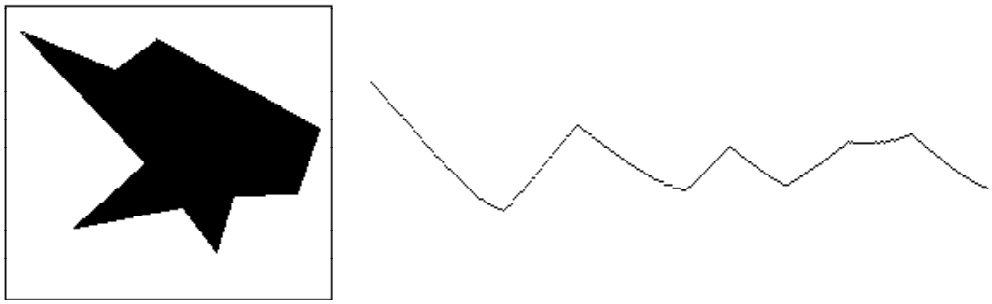

Fig. 9.2: Semnătura unei forme poligonale în funcţie de abscisa curbilinie; punctul de referinţă este centrul de greutate, axa de referinţă este orizontala.

unde $[\mathbf{xy}]$ semnifică segmentul de dreaptă definit de punctele $\mathbf{x}$ şi $\mathbf{y}$. În cazul formelor concave, nucleul formei este o mulţime vidă.

Folosind semnătura formei, se pot defini parametri de tip geometric. Raportul de simetrie este definit ca

$$Y(K) = \sup_{\mathbf{x}\in\mathbf{K}} \inf_{\theta\in[0;\pi]} \frac{d_{\mathbf{x}}(\theta)}{d_{\mathbf{x}}(\theta+\pi)} \tag{9.18}$$

Se observă că nu s-a făcut nici o presupunere privind convexitatea formei K, dar unicitatea distanţelor $d_{\mathbf{x}}(\theta)$ şi $d_{\mathbf{x}}(\theta+\pi)$ implică calcularea raportului de simetrie după punctele ce aparţin nucleului formei.

Raportul de circularitate este definit ca:

$$C(K) = \frac{\sup_\theta \left(d_{\mathbf{x}}(\theta) + d_{\mathbf{x}}(\theta+\pi)\right)}{\inf_\theta \left(d_{\mathbf{x}}(\theta) + d_{\mathbf{x}}(\theta+\pi)\right)} \tag{9.19}$$

Funcţia $h_K(\theta) = d_{\mathbf{x}}(\theta) + d_{\mathbf{x}}(\theta+\pi)$ se numeşte suportul formei, şi este diametrul formei pe direcţia θ.

În [5] s-a propus aproximarea formelor prin dezvoltarea în serie Fourier a semnăturii acestora şi truncherea reprezentării (tehnică utilizată şi într-o aplicaţie clasică de recunoaştere a conturului unor tipuri de avioane). O altă posibilă aplicaţie a semnăturii pleacă de la observaţia că, pentru o formă poligonală, în semnătură, poziţia vârfurilor este marcată de puncte unghiulare (a se urmări figurile 9.1 şi 9.2). Atunci aceasta poate fi o metodă de aproximare poligonală a unei forme oarecari (găsirea vârfurilor unui poligon ce o aproximează cât mai bine).

9.4 Skeletoane morfologice şi generalizate

Skeletonul este o reprezentare bidimensională simplificată, echivalentă, a unei forme. Pentru o formă A oarecare se defineşte discul maximal în A, de centru $\mathbf{x}$ şi rază r, $B_{\mathbf{x}}(r)$ ca fiind discul caracterizat de:

$$B_{\mathbf{x}}(r) \subseteq A \tag{9.20}$$

$$B_{\mathbf{x}}(r) \subseteq B_{\mathbf{x}'}(r') \subseteq A \Leftrightarrow \begin{cases} r = r' \\ \mathbf{x} = \mathbf{x}' \end{cases} \tag{9.21}$$

Deci discul maximal trebuie să fie inclus în formă (9.20) şi nici un alt disc inclus în formă să nu îl includă, sau, dacă îl include, să fie identic cu acesta (9.21). O formă poate avea mai multe discuri maximale.

Skeletonul unei forme este mulţimea centrelor discurilor maximale ale formei. Ca exemple simple, skeletonul unui disc este centrul său, skeletonul unui pătrat este reuniunea diagonalelor sale (figura 9.3).

Fig. 9.3: Exemple de skeletoane ale unor forme continue simple.

9.4.1 Skeletonul morfologic

Calculul skeletonului unei forme reprezentate în spaţiul discret se poate face prin formula Lantuejoul [15]; skeletonul formei A, $SK(A)$, este format din reuniunea unui număr finit de seturi skeleton:

$$SK(A) = \bigcup_{n=0}^{N_{\max}} S_n(A) \tag{9.22}$$

$$S_n(A) = (A \ominus nB) - (A \ominus nB) \circ B \tag{9.23}$$

Ordinul $N_{\max}$ corespunde momentului în care toate seturile skeleton succesive devin nule, moment marcat de $A \ominus N_{\max}B = \emptyset$; elementul structurant nB este iterarea de n ori a elementului structurant B, $nB = B \oplus ... \oplus B$ (de n ori). Elementul structurant folosit

este în general o expresie discretă a discului unitar (deci element structurant V_4 sau V_8). Figura 9.4 prezintă skeletonul unei forme discrete oarecare.

Fig. 9.4: Skeleton al unei forme discrete, reprezentat prin reuniunea seturilor skeleton şi informaţia de apartenenţă a fiecărui punct la un anumit set skeleton.

Reconstrucţia formei din skeleton se face după formula

$$A = \bigcup_{n=0}^{N_{\max}} S_n(A) \oplus nB \tag{9.24}$$

Se pot realiza şi reconstrucţii parţiale (aproximări ale mulţimii A) prin neglijarea seturilor skeleton ce reprezintă detaliile (acestea sunt seturile skeleton de ordin mic); aproximarea de ordin k a formei înseamnă deci ignorarea primelor k seturi skeleton din reconstrucţie:

$$\widetilde{A_k} = \bigcup_{n=k}^{N_{\max}} S_n(A) \oplus nB \tag{9.25}$$

Skeletonul este invariant la translaţie, nu este conex (chiar dacă forma A este conexă; a se vedea figura 9.5), nu este comutativ cu operaţia de reuniune a formelor (a se vedea figura 9.6). Transformarea skeleton este idempotentă ($SK(SK(A)) = SK(A)$) şi antiextensivă ($SK(A) \subseteq A$). Seturile skeleton sunt disjuncte două câte două ($S_i(A) \cap S_j(A) = \emptyset$, $\forall i \neq j$).

Folosirea skeletonului morfologic pentru recunoaşterea fomelor este restricţionată de puternica sa sensibilitate la zgomote (o mică schimbare a formei duce la o modificare semnificativă a skeletonului[1]), şi de variaţia la rotaţia şi scalarea obiectelor (pentru reprezentări în spaţiul discret). În acelaşi timp, folosirea elementului structurant de tip disc unitar aduce o puternică dependenţă de metrica folosită în definirea sa (să nu uităm că într-un

[1] Exemplul clasic este de a considera un disc fără centru; în acest caz skeletonul este o coroană circulară situată la jumătatea razei.

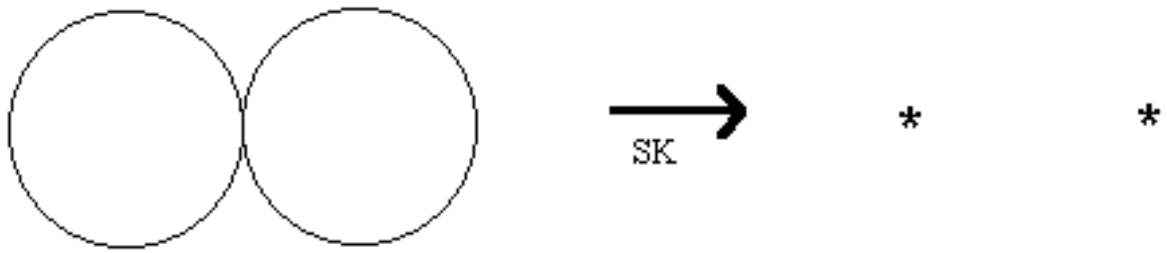

Fig. 9.5: Skeletonul nu este conex.

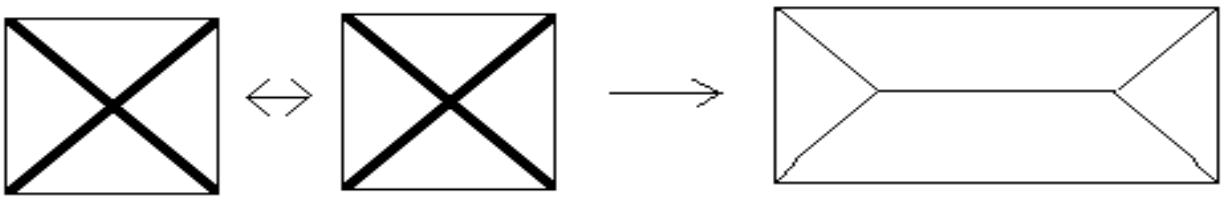

Fig. 9.6: Transformata skeleton nu este comutativă cu reuniunea.

spaţiu discret, metrica Euclidiană nu este cea mai favorabilă) şi produce elemente structurante pătrate (V_8) sau în cruce (V_4), ce pot fi cu greu interpretate ca discuri. Aceasta a dus la folosirea unei clase de elemente structurante care să nu provină din noţiunea de metrică - elementele structurante generalizate.

9.4.2 Skeletonul generalizat

Fie $\{G_i\}$ un set de mulţimi, numit set generator. Elementele structurante generalizate sunt definite recurent prin:

$$B_i = B_{i-1} \oplus G_i,\ i = 1, 2, ... \tag{9.26}$$

$$B_0 = 0_n$$

În general se folosesc seturi generatoare având o anumită periodicitate T ($G_{i+kT} = G_i,\ \forall i, k \in \mathbf{Z}$). De exemplu, figura 9.7 prezintă construcţia setului de elemente structurante generalizate rezultat dintr-un set generator cu perioadă 1 (deci compus dintr-o singură mulţime), $E_1 = \{(-1,-1), (-1,0), (0,-1), (0,0)\}$.

Fiecărui punct al formei A i se va asocia ordinul (indicele) elementului structurant generalizat maximal centrat cu originea în punctul respectiv, construind astfel o "hartă de

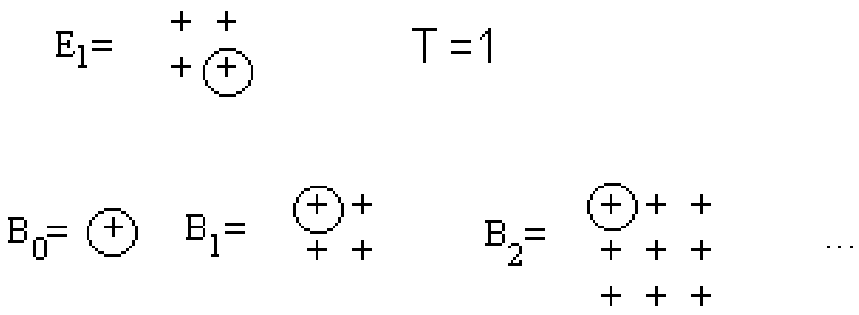

Fig. 9.7: Construcţia unui set de elemente structurante generalizate.

distanţe" (a se vedea şi figura 9.8):

$$D(\mathbf{x}) = \begin{cases} 0, \text{ dacă } \mathbf{x} \notin A \\ n, \text{ dacă } (B_{n-1})_{\mathbf{x}} \subseteq A \text{ şi } (B_n)_{\mathbf{x}} \nsubseteq A \end{cases} \tag{9.27}$$

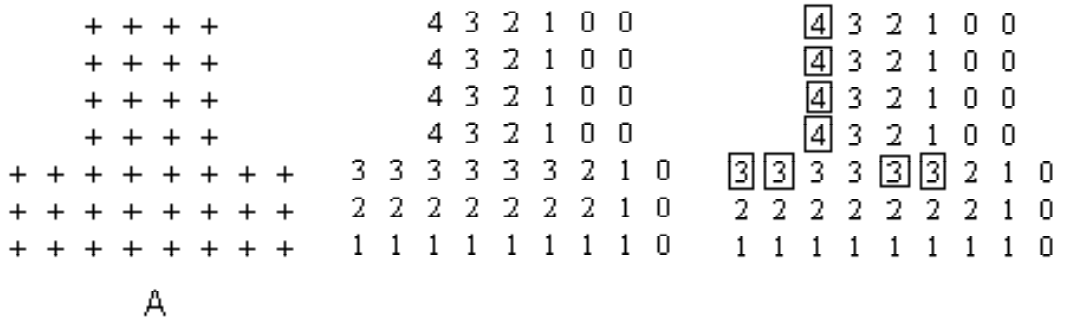

Fig. 9.8: Harta de distanţe a unui obiect construită pe baza setului de elemente structurante generalizate prezentat anterior şi skeletonul generalizat corespunzător.

Skeletonul generalizat al unei forme este mulţimea originilor elementelor structurante generalizate maximale în formă[2]:

$$GSK(A) = \left\{ \mathbf{x} \in A | \left(B_{D(\mathbf{x})-1}\right)_{\mathbf{x}} \nsubseteq \left(B_{D(\mathbf{y})-1}\right)_{\mathbf{y}}, \forall \mathbf{x} \neq \mathbf{y} \right\} \tag{9.28}$$

Folosind harta de distanţă anterioară, skeletonul generalizat obţinut este prezentat în figura 9.8.

[2]Se poate remarca similitudinea cu definiţia skeletonului, în care s-au înlocuit noţiunile: centru al discului cu origine a elementului structurant, disc cu element structurant generalizat, rază a discului cu indice (ordin) al elementului structurant generalizat.

Capitolul 10

PRINCIPII DE IMPLEMENTARE SOFTWARE ŞI HARDWARE

Principiile esenţiale legate de implementările practice ale sistemelor de prelucrarea şi analiza imaginilor urmăresc două direcţii, cu dezvoltare corelată: implementările software şi dispozitivele hardware (de accelerare). După cum am arătat în capitolul introductiv, la descrierea structurii tipice a unui sistem de prelucrarea şi analiza imaginilor, marea majoritate a implementărilor folosesc ca suport fizic pentru unitatea centrală de prelucrare un calculator obişnuit (compatibil PC); ceea ce îl particularizează este pachetul de programe rulate. Putem distinge două categorii fundamentale de astfel de programe: programe strict dependente de aplicaţie şi programe de uz general.

Un program dependent de aplicaţie realizează doar algoritmii specifici operaţiei pe care o execută sau supraveghează. Interacţiunea cu operatorul uman este minimă şi calificarea acestuia nu este necesar să o depăşească pe cea a unui tehnician [11]. Adeseori programul trebuie să fie de timp real. Aceste variante de implementare sunt potrivite pentru procese caracterizate de parametri stabili şi care se desfăşoară în condiţii ambiante (iluminare, poluare vizibilă – particule, fum) relativ constante şi nu sunt portabile (fiind în general optimizate pentru o anumită structură hardware).

Programele de uz general permit efectuarea unui mare număr de operaţii de prelucrarea şi analiza imaginilor, cu numeroşi parametri reglabili. Interacţiunea cu operatorul uman este mare şi acesta trebuie să aibă o calificare superioară. Programele de acest tip nu sunt de timp real şi în general sunt cuplate *off-line* cu instalaţiile tehnologice propriuzise, făcând parte mai ales din dotarea laboratoarelor de cercetare şi de analiza calităţii şi conformităţii. Interfaţa utilizator este cea care crează diferenţa între două categorii de programe, în ceea ce priveşte modalitatea în care operatorul specifică succesiunea de operaţii de executat. Din acest punct de vedere vom face distincţia între *menu-driven* şi *flow-chart driven* (deci programe controlate prin meniu sau prin graf de flux).

Implementările de tip *menu-driven* aplică câte un unic pas de prelucrare asupra imaginii din fereastra activă; rezultatul va fi prezentat într-o nouă fereastră de afişare. Tipurile de operaţii se aleg din meniuri sau bara de butoane. Asemenea soluţii corespund majorităţii sistemelor software comerciale de grafică şi *imaging* (de tipul Adobe Photoshop, Corel Draw, Paint Shop Pro). Operaţiile realizate sunt orientate mai ales către aspectul grafic (publicistic) al imaginilor, referindu-se în special la operaţii de filtrare, modificare a contrastului, pseudocolorare, decupare de regiuni. Programele cu comandă *menu-driven*, cu rezultatele intermediare parcurgând etapele operaţiilor din fereastră în fereastră, sunt direct derivate din proiectarea obiectelor de interfaţă în sistemele de tip Windows.

Implementările de tip *flow-chart* permit construirea grafică interactivă a unui lanţ de operaţii aplicate unei imagini iniţiale. Fiecare operaţie este un bloc funcţional caracterizat de intrări, ieşiri şi parametri de control; blocurile funcţionale sunt interconectate, implementând fluxul de operaţii pe care îl va parcurge imaginea. Procesul de comandă înseamnă selectarea imaginii iniţiale şi acţionarea unui buton de start. Asemenea programe sunt tipice sistemelor de calcul mari (staţii de lucru), pentru care produsul Khoros este un standard de fapt; un produs similar pentru PC este programul AdOculos (figura 10.1).

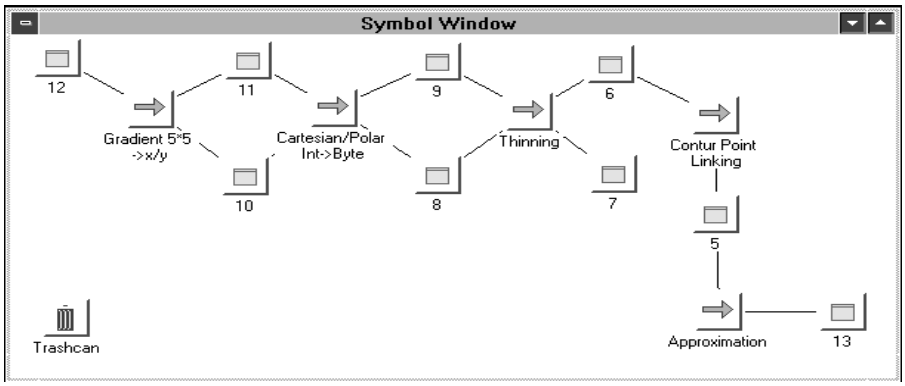

Fig. 10.1: Specificarea fluxului de operaţii în fereastra de comandă a programului AdOculos.

Ca o categorie intermediară între comenzile meniu şi diagrama de flux, putem considera programele cu comandă de tip macro (sau *batch*); programul (sau limbajul) Matlab poate fi încadrat într-o asemenea categorie.

Multe dintre operţiile iniţial realizate prin software-ul de aplicaţie au migrat către nivelul hardware. Exemplul cel mai tipic de asemenea comportament este dat de rezolvare problemei de dithering (aproximarea culorilor reale dintr-o imagine. folosind un număr mic de culori disponibile la dispozitivul de afişaj). După ce, iniţial, problema era rezolvată

de programele aplicaţie, gestiunea culorilor a trecut în grija sistemului de operare, pentru ca, recent, să apară plăcile grafice inteligente. Aceasta este de altfel una dintre tendinţele actuale: evoluţia accesoriilor (plăci specializate, dispozitive de achiziţie) inteligente, deci cu facilităţi de prelucrare integrate. Senzorii de imagine CCD sunt una dintre ţintele predilecte ale acestei dezvoltări, înglobând cipuri de compresie şi transmisie a imaginilor, comandă de urmărire automată a ţintei, operaţii de recunoaştere.

Deşi programarea orientată pe obiecte este un concept destul de nou, evoluţia sa a fost deosebit de spectaculoasă. Probabil că principalul atu al acestei evoluţii a fost apariţia sistemului de operare Windows (cu toate derivatele sale 3.1, 3.11, '95, '98, etc.) a cărui structură este în întregime bazată pe conceptul de obiecte. Programarea unei aplicaţii Windows va face în mod clar apel la structuri predefinite - clasele de bază aplicaţie, fereastră, buton, ş.a.m.d. Atunci devine destul de normal să fie proiectate şi datele specifice ale aplicaţiei particulare tot ca structuri de tip obiecte.

Alegerea unui limbaj specific de programare nu mai este totuşi o problemă esenţială; majoritatea limbajelor de uz general ingineresc (tehnic) provin din trunchiul comun C / Pascal. Desigur că există nenumărate variante ale acestor limbaje de bază (C++ Builder, Visual C++, Delphi, Borland C++ şi Borland Pascal cu obiecte) şi chiar apar limbaje noi (Java, care, deşi se revendică ca o aplicaţie pur distribuită pentru folosirea reţelelor şi în special a Internetului, nu este cu mult diferită ca structură de un C cu clase). Principala evoluţie a mediilor de programare nu a fost însă legată de modificarea limbajului în sine (şi deci a caracteristicilor de compilare a codului) ci modificarea stilului în care se crează aplicaţia specifică, şi mai precis, aspectul de interfaţă.

Proiectarea unei interfeţe pentru Windows înseamnă definirea unor acţiuni asociate evenimentelor produse în sistem (clicuri de mouse, apăsări de taste) şi construirea elementelor grafice specifice (ferestre de afişare şi de dialog, meniuri, introducere date, etc.). Acestă activitate a fost simplificată la maximum prin apariţia constructoarelor vizuale de aplicaţii, în care interfaţa grafică este construită prin alipirea unor elemente de bază (butoane, ferestre, zone de text) pe cadre fixe (fereastră), numai cu mouse-ul (prin tehnică *drag and drop*). Compilatorul generează automat codul sursă aferent construcţiei; tot ceea ce trebuie să facă programatorul este să lege sursa ce descrie aplicaţia şi acţiunea specifică la codul care interpretează evenimentul de apel.

Ceea ce trebuie subliniat (spre a evita anumite interpretări uzuale, dar eronate) este că aceste medii de programare avansate (gen Builder, Visual, etc..) nu preiau şi atribuţia de a scrie algoritmii specifici; uşurinţa construcţiei vizuale este legată strict de construcţia interfeţei aplicaţiei, deci aspect grafic şi eventual partea de preluare a datelor. Programatorul va dezvolta restul aplicaţiei ca pentru un compilator clasic.

Concluzia ce rezultă din această scurtă discuţie este aceea că, pentru orice aplicaţie, *trebuie separată partea de interfaţă de partea de calcul specific.* În cazul unui program de prelucrarea şi analiza imaginilor acesta însemnă că trebuie făcută o proiectare la nivelul structurii de date (matricea ce conţine valorile pixelilor din imagine şi prelucrările specifice

asupra acestora) şi o proiectare la nivelul interfeţei (care să specifice cum se va face afişarea imaginilor pe ecran, cum vor fi scrise pe disc, cum se va face un eventual transfer dinamic al datelor de la sau către alte aplicaţii). Majoritatea covârşitoare a sarcinilor legate de interfaţă pot fi rezolvate fără a cunoaşte nimic despre operaţiile specifice prelucrării imaginilor şi despre modul în care o imagine este reprezentată în memoria de lucru a calculatorului. Să considerăm de exemplu problema afişării unei imagini într-o fereastră. Fără îndoială că cel mai simplu mod de afişare este folosirea imaginii ca un *canvas*, înglobat în obiectul de tip fereastră, şi de a cărui afişare se ocupă sistemul Windows. Această abordare exclude însă accesul la conţinutul imaginii; aceasta trebuie deci separată de fereastra de afişare. Este poate deci preferabilă memorarea imaginii ca o matrice şi transformarea acesteia într-o structură *bitmap* ataşată ferestrei, la fiecare cerere de reafişare a acesteia.

Cuvântul de ordine actual în implementările software este orientarea-obiect (limbajele C++, Delphi, Java), insistând mai ales pe aspectele de polimorfism şi moştenire pe care le aduce acest stil de programare. Moştenirea corespunde cazului unor imagini cu structură din ce în ce mai elaborată, pentru care se pot face tot mai multe operaţii (de tip analiză de formă şi clasificare, de exemplu). Polimorfismul corespunde realizării unor operaţii cu nume (sau metode de implementare) identice care să aibă comportări diferite, în funcţie de tipul datelor cărora li se aplică (funcţia de histogramă poate produce, de exemplu, funcţia de densitate de probabilitate a câmpului aleator imagine, în cazul imaginilor cu nivele de gri, aria obiectelor, în cazul imaginilor binare şi numărul de componente conexe, în cazul imaginilor binare etichetate). De asemenea, aspectul legat de refolosirea codului nu este neglijabil (să nu uităm, că, de exemplu, pentru filtrarea în domeniul spaţial, un numitor comun al diverselor tipuri de filtre este tehnica de tip fereastră glisantă).

Tehnologia Java, aduce, pa lângă orientarea obiect, ideea de folosire a resurselor distribuite (fie pe Internet, fie ca resurse de multiprocesare). Astfel a devenit posibilă crearea de depozite de software specializat, universal executabil (prin mecanismul de *applet*) pe orice maşină pe care este disponibil un *browser* Internet. Marile avantaje anunţate pentru Java (cod portabil, execuţie paralelă, protecţie la manipularea defectuoasă a zonelor de memorie, protecţia datelor) sunt contrabalansate de codul executabil relativ lent şi de dimensiunile mari ale codului sursă.

Java este însă interesantă pentru prelucrarea şi analiza imaginilor prin perspectiva deschisă de *multi-threding* – existenţa şi gestionarea de către o aplicaţie oarecare a mai multe fire de execuţie ce rulează concomitent din punct de vedere logic (sau aproape concomitent din punctul de vedere fizic al calculatorului cu unic procesor); astfel devine posibilă rularea paralelă de aplicaţii pe sisteme cu unic procesor, fără sisteme de operare *multi-tasking*. De altfel, puţine sisteme fizice multiprocesor au fost realizate pentru prelucrarea de imagini; sistemele de prelucrarea şi analiza imaginilor au utilizat în general maşini cu paralelism masiv (SIMD/ SPMD, *transputer farm*) sau grupuri de calculatoare (*cluster processing*).

Exploatarea paralelismului este un merit pe care prelucrarea imaginilor şi l-a arogat încă de la începuturi; operaţiile tipice de prelucrarea imaginilor sunt operaţii relativ simple, cu un număr mare de instanţe (pentru fiecare pixel al imaginii se face ceva), folosind date puţin redundante (suprapunerea între poziţiile alăturate ale ferestrelor de filtrare este în general mică) şi, uneori, informaţie globală (cazul histogramei sau a filtrărilor în domeniul de frecvenţă). Se pot distinge astfel două nivele de paralelism: un paralelism masiv, intrinsec structurii imaginii, legat de nivelul pixel, şi un paralelism ascuns, specific operaţiilor de prelucrare (ca de exemplu realizarea operaţiilor morfologice prin (6.3) şi (6.6) sau implementarea paralelă a FFT).

În concluzie, putem afirma că prelucrarea şi analiza imaginilor a reuşit să câştige teren ca urmare a realizărilor spectaculoase ale tehnologiei electronice şi informaticii. Creşterea continuă a puterii de calcul disponibile în unităţile de prelucrare ale calculatoarelor va transforma probabil în viitorul apropiat prelucrarea şi analiza imaginilor dintr-o anexă nebuloasă şi exotică a aplicaţiilor speciale într-o soluţie fiabilă de larg consum industrial.

Bibliografie

[1] Buzuloiu, V.: *Prelucrarea imaginilor: note de curs*, Universitatea "Politehnica" Bucureşti, 1998

[2] Castleman, K. R.: *Digital Image Processing*, Prentice Hall, Englewood Cliffs, NJ, 1996

[3] Cocquerez, J. P., Philipp, S. (coord.): *Analyse d'images: filtrage et segmentation*, Masson, Paris, 1995

[4] Dougherty E. R., Giardina, C. R.: *Image Processing - Continous to Discrete*, vol. 1, *Geometric, Transform and Statistical Methods*, Prentice Hall Inc., Englewood Cliffs, 1987

[5] Gonzales, R. C., Woods, R. E.: *Digital Image Processing*, Addison Wesley, Reading MA, 1992

[6] Haralick, R. M., Shapiro, L. G.: *"Glossary of Computer Vision Terms"*, în Pattern Recognition, vol. 24, no. 1, pag. 69-93, 1991

[7] Haralick, R. M., Sternberg, S. R., Zhuang, X.: "*Image Analysis using Mathematical Morphology*", în IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 9, no. 4, Iulie 1987, pag. 532-549

[8] Jähne, B.: *Practical Handbook on Image Processing for Scientific Applications*, CRC Press, 1997

[9] Jain, A. K.: *Fundamentals of Digital Image Processing*, Prentice Hall, Englewood Cliffs NJ, 1989

[10] Ministerium für Wirtschaft, Mittelstand und Technologie des Landes Nordrhein-Westfalen: *Stand und Trends der Bidverarbeitung in NRW*, Düsseldorf, 1995

[11] Ministerium für Wirtschaft, Mittelstand und Technologie des Landes Nordrhein-Westfalen: *Produkte und Dienstleitungen für die Bildverarbeitung. Stand und Trends*, Düsseldorf, 1996

[12] Pitas, I., Venetsanopoulos, A. N.: *Nonlinear Digital Filters – Principles and Applications*, Kluwer Academic Publ., Norwell MA, 1990

[13] Press, W. H., Flannery, B. P., Teukolsky, W. T., Vetterling, W. T.: *Numerical Recipes in C. The art of scientific computing*, Cambridge University Press, 1988

[14] Serra, J.: *Image Analysis and Mathematical Morphology*, Academic Press, London, 1982

[15] Schmitt, M., Mattioli, J.: "*Reconnaissance de formes planaires par morphologie mathematique et reseaux de neurones*", în Revue Technique Thomson CSF, vol. 22, no. 4, Decembrie 1990, pag. 573-609, Ed. Gauthiers-Villars: Paris

[16] Spătaru, A.: *Teoria Transmisiunii Informaţiei*, Ed. Didactică şi Pedagogică, Bucureşti, 1984

[17] Vertan, C., Gavăt, I., Stoian, R.: *Variabile aleatoare: principii şi aplicaţii*, Editura Printech, Bucureşti, 1999

[18] Zamperoni, P.: "*Image Enhancement*", Advances in Imaging and Electron Physics, vol. 92, pp. 1-77, Academic Press, 1995

[19] Wahl, F. M.: *Digital Image Signal Processing*, Artech House, Boston, 1987