# 4. Trăsături geometrice ale obiectelor binare

## 4.1. Introducere

În această lucrare sunt prezentate câteva trăsături importante ale obiectelor binare precum şi modul de calcul al acestora. Trăsăturile prezentate sunt aria, centrul de masă, axa de alungire, perimetrul, factorul de subțiere al obiectului, elongația şi proiecțiile obiectului binar.

## 4.2. Considerații teoretice

În urma operațiilor de segmentare şi etichetare a obiectelor din imaginile digitale se obține o imagine cu obiecte care pot fi referite distinct.

Obiectul 'i' poate fi descris de următoarea funcție:

$$I_i(r,c) = \begin{cases} 1, & \text{daca } I(r,c) \in \text{obiectului etichetat 'i'} \\ 0 & \text{in celelalte cazuri} \end{cases}$$

Trăsăturile geometrice ale obiectelor se pot clasifica în următoarele două categorii:

- de poziționare şi orientare: centrul de masă, aria, perimetrul, axa de alungire;
- de formă: factorul de aspect, factorul de subțiere, numărul Euler, proiecțiile, diametrele Feret ale obiectelor

### 4.2.1. Aria

$$A_i = \sum_{r=0}^{N-1}\sum_{c=0}^{N-1} I_i(r,c)$$

Aria $A_i$ este măsurată în pixeli şi indică mărimea relativă a obiectului.

### 4.2.2. Centrul de masă

$$\bar{r}_i = \frac{1}{A_i}\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} r I_i(r,c)$$

$$\bar{c}_i = \frac{1}{A_i}\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} c I_i(r,c)$$

Aceste formule corespund liniei şi respectiv coloanei centrului obiectului. Această proprietate ajută la localizarea obiectului într-o imagine bidimensională.

### 4.2.3. Axa de alungire (axa de inerție minimă / momentul de inerție de ordin 2)

$$\tan(2\phi_i) = 2\frac{\sum_{r=0}^{N-1}\sum_{c=0}^{N-1}(r-\bar{r}_i)(c-\bar{c}_i)I_i(r,c)}{\sum_{r=0}^{N-1}\sum_{c=0}^{N-1}(c-\bar{c}_i)^2 I_{i(r,c)} - \sum_{r=0}^{N-1}\sum_{c=0}^{N-1}(r-\bar{r}_i)^2 I_i(r,c)}$$

Dacă atât numărătorul cât şi numitorul fracţiei din formula de mai sus sunt zero atunci obiectul prezintă simetrie circulară, orice dreaptă ce trece prin centrul de greutate fiind axă de simetrie.

Astfel, pentru aflarea unghiului trebuie aplicată funcţia arctangentă. Această funcţie este definită pe domeniul $(-\infty, +\infty)$ şi are valori în domeniul $(-\pi/2, \pi/2)$. Evaluarea acestei funcţii devine instabilă atunci când numitorul fracţiei se apropie de 0. Totodată, semnele numitorului şi numărătorului sunt importante pentru determinarea cadranului corect în care se află rezultatul, dar funcţia arctangentă nu face diferenţa între direcţii diametral opuse. Din acest motiv se recomandă folosirea funcţiei „atan2", funcţie care ia ca argumente numărătorul şi respectiv numitorul unei astfel de fracţii, şi returnează un rezultat în domeniul $(-\pi, \pi)$.

Această proprietate dă informaţii despre orientarea obiectului. Axa corespunde direcţiei în jurul căreia obiectul (văzut ca o suprafaţă plană de grosime constantă) se poate roti cel mai uşor (are momentul cinetic minim).

### 4.2.4. Perimetrul

Perimetrul obiectului ne ajută să determinăm poziţia obiectului în spaţiu şi detalii despre forma lui. Perimetrul poate fi determinat prin numărarea pixelilor de pe contur (pixeli cu valoarea 1 care au cel puţin un pixel vecin de valoare 0).

O primă variantă de detecţie a conturului ar fi scanarea imaginii, linie cu linie, şi numărarea pixelilor din obiect, care îndeplinesc condiţia mai sus menţionată. Ca principal dezavantaj al acestei metode este faptul că nu putem face distincţie între conturul exterior şi eventualele contururi interioare (datorate găurilor din obiect). Deoarece pixelii din imaginile digitale sunt distribuiţi după un rastru dreptunghiular, lungimile curbelor şi ale dreptelor oblice nu pot fi estimate corect prin numărarea pixelilor. O primă corecţie ar fi înmulţirea perimetrului rezultat la algoritmul precedent cu $\pi/4$. Există şi alte metode de corecţie a lungimilor care ţin seama de tipul de vecinătate folosit (V4, V8 etc.)

Altă metodă de detecţie a conturului unui obiect ar fi detecţia muchiilor sale folosind un algoritm de detecţie consacrat, subţierea acestora la grosime unitară şi numărarea pixelilor de muchie rezultaţi.

Metodele de tip "chain-codes" sunt metode mai complexe de detecţie a conturului şi oferă acurateţe mai mare.

### 4.2.5. Factorul de subţiere al obiectului (thinness ratio)

$$T = 4\pi\left(\frac{A}{P^2}\right)$$

Această funcţie are valoarea maximă 1, care corespunde cercului. Această proprietate poate fi folosită pentru a vedea cât de rotund este un obiect. Cu cât este mai aproape de 1 cu atât obiectul este mai rotund. Această valoare dă şi informaţii despre regularitatea obiectului. Obiectele cu contur regulat au o valoare a raportului mai mare decât obiectele cu contur neregulat. Valoarea 1/T se numeşte factor de neregularite al obiectului (sau factor de compactitate).

### 4.2.6. Elongația (factorul de aspect) (“aspect ratio”/elongație/excentricitate)

Această mărime se determină prin scanarea imaginii şi determinarea valorilor minime şi maxime ale liniilor şi coloanelor ce formează dreptunghiul circumscris obiectului.

$$R = \frac{c_{\max} - c_{\min} + 1}{r_{\max} - r_{\min} + 1}$$

### 4.2.7. Proiecțiile obiectului binar

Proiecțiile ne dau informații despre forma obiectului. Proiecția orizontală este egală cu suma pixelilor pe linii, iar proiecția verticală este egală cu suma pixelilor pe coloane.

$$h_i(r) = \sum_{c=0}^{N-1} I_i(r,c) \qquad v_i(c) = \sum_{r=0}^{N-1} I_i(r,c)$$

Proiecția este utilă în programele de recunoaştere a scrisului unde obiectul care ne interesează poate fi normalizat.

## 4.3. Detalii de implementare

Pentru a face distincție între diversele obiecte prezente în imagine vom considera că fiecare dintre ele este desenat cu o culoare diferită. Aceste culori pot fi rezultatul unei operații prealabile de etichetare sau pot fi generate manual (vezi Fig. 4.1).

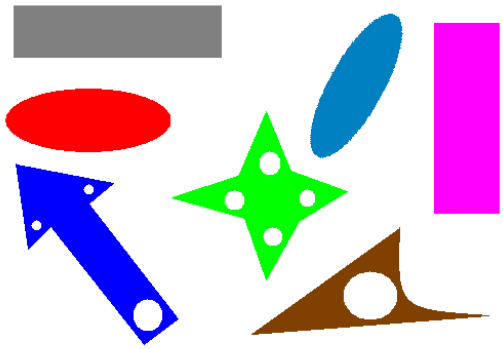

**Fig. 4.1** Exemplu de imagine etichetată pe care pot fi testați algoritmii descrişi aici

Există mai multe abordări pentru implementarea extragerii proprietăților geometrice. O abordare simplă este calcularea lor pentru toate obiectele ca o singură operație. Într-o imagine în formatul descris mai sus pot exista maximum 255 de obiecte+fundalul, fiecare dintre ele având o culoare diferită.

O a doua abordare posibilă constă în selecția cu mouse-ul a obiectelor. Utilizatorul va poziționa cursorul mouse-ului deasupra unui pixel aparținând obiectului şi va efectua dublu-click. Ca răspuns la această acțiune se va afişa o fereastră de mesaj în care se vor afişa trăsăturile geometrice dorite.

Pentru a adăuga o metodă handler pentru evenimentul de dublu click se vor parcurge următorii paşi (vezi Fig. 4.2):

1. Selectați tabulatorul *Class View*;
2. Efectuați click dreapta pe clasa *CDibView* şi apoi *Properties…*;
3. În fereastra *Properties* se alege secțiunea *Messages*;
4. La mesajul *WM_LBUTTONDBLCLK*, din lista derulantă, se adaugă metoda *OnLButtonDblClk* (*<Add> OnLButtonDblClk*). Wizard-ul va genera o metodă handler pentru mesajul respectiv, care va primi ca parametru coordonatele cursorului mouse-ului relativ la colțul din stânga sus al zonei client a ferestrei view, la momentul când a fost efectuat dublu-click.

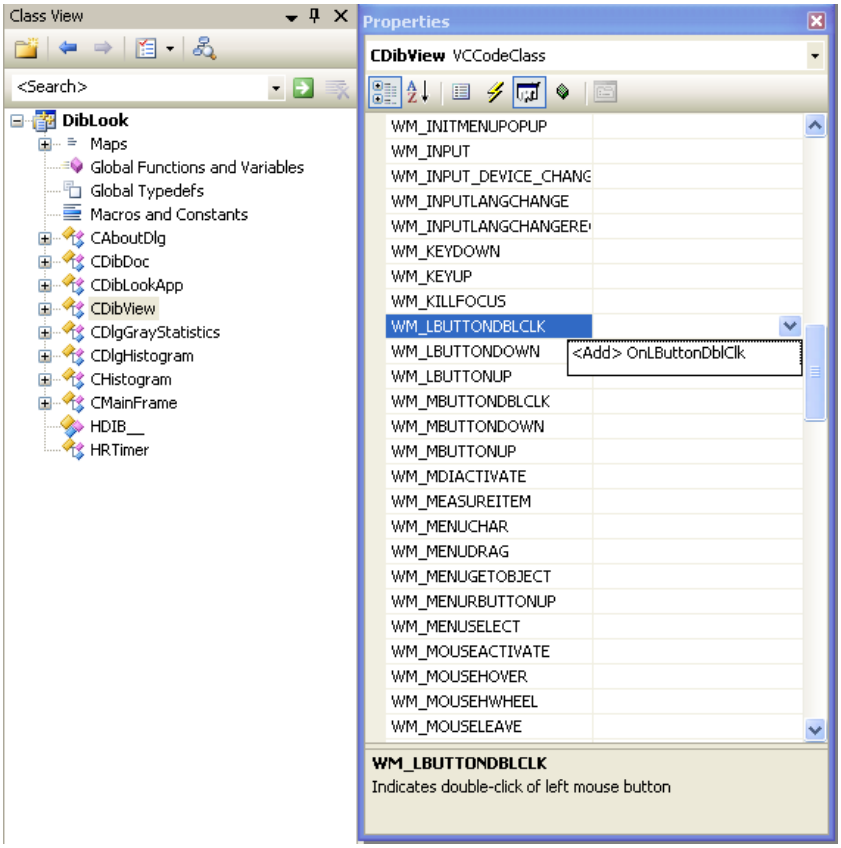


**Fig. 4.2** Adăugarea unui nou handler de mouse pentru evenimentul de dublu-click stânga clasei *CDibView*

Următoarea secțiune de cod prezintă un handler simplu pentru evenimentul de dublu click care calculează coordonatele cursorului mouse-ului relativ la colțul stânga jos al imaginii şi afişează o fereastră de mesaj conținând coordonatele şi indexul din paletă al pixelului deasupra căruia s-a efectuat dublu click.

```
void CDibView::OnLButtonDblClk(UINT nFlags, CPoint point)
{
     BEGIN_SOURCE_PROCESSING;

     //obține poziția curentă de scroll (din cauza barelor de scroll
     //coordonatele pot fi translatate)şi ajustează coordonatele
     CPoint pos = GetScrollPosition()+point;

     //variabila pos conține coordonatele relativ la colțul stânga sus al
     zonei client a view-ului
     //axa y trebuie inversată din cauza reprezentării bitmap-urilor în
     memorie
     int x = pos.x;
     int y = dwHeight-pos.y-1;
```

```
     //testează dacă punctul este poziționat în interiorul imaginii
     if (x>0 && x<dwWidth && y>0 && y<dwHeight)
     {
          //pregăteşte un obiect CString pentru formatarea mesajului
          CString info;
          info.Format("x=%d, y=%d, color=%d", x, y, lpSrc[y*w+x]);
          AfxMessageBox(info);
     }

     END_SOURCE_PROCESSING;

     //apelarea metodei superclasei
     CScrollView::OnLButtonDblClk(nFlags, point);
}
```

O a treia abordare posibilă constă în selecția obiectelor individuale sau calcularea proprietăților tuturor obiectelor urmată de afişarea rezultatelor direct pe imaginea destinație. Un mod convenabil de a afişa text şi elemente grafice într-o imagine este folosirea funcțiilor Windows GDI (Graphics Device Interface).

Fiecare operație grafică din Windows se efectuează folosind un obiect DC (Device Context). Acest obiect conține date precum device-driver-ul care efectuează operațiile de desenare (driver-ul plăcii video, al imprimantei, driver-ului unui DC aflat in memorie), suprafața pe care se efectuează desenarea (suprafața asociată display-ului, o suprafață secundară, o suprafață aflată în memorie sistem), obiectul pen folosit la desenare (culoare, lățimea liniei), obiectul brush folosit pentru umplerea obiectelor şi aşa mai departe.

Pentru a desena peste un bitmap, acesta trebuie selectat într-un obiect DC, astfel încât toate operațiile de desenare ulterioare se vor efectua pe suprafața acestuia. Bitmap-urile device independent (DIBs) folosite de către DIBView nu pot fi selectate direct într-un DC. Pentru a ocoli această problemă se poate crea un bitmap device dependent (DDB) şi inițializat cu pixelii imaginii sursă. Apoi acest bitmap va fi selectat într-un DC (aflat în memorie) şi se vor efectua operațiile de desenare. În final, pixelii din DDB se vor copia înapoi în DIB-ul destinație.

Următoarea secțiune de cod parcurge toți aceşti paşi, şi afişează o linie şi un text la coordonatele din imagine unde a fost efectuat dublu-click-ul cu mouse-ul.

```
void CDibView::OnLButtonDblClk(UINT nFlags, CPoint point)
{
     BEGIN_PROCESSING();

     CDC dc; //obiectul DC-ul aflat în memorie
     //se crează astfel încât să fie compatibil cu ecranul
     dc.CreateCompatibleDC(0);

     //va conține un DDB compatibil cu ecranul
     CBitmap ddBitmap;

     //creare DDB compatibil cu ecranul şi inițializarea datelor acestuia
     //cu datele din imagine DIB sursă
     HBITMAP hDDBitmap =
     CreateDIBitmap(::GetDC(0), &((LPBITMAPINFO)lpS)->bmiHeader, CBM_INIT,
     lpSrc, (LPBITMAPINFO)lpS, DIB_RGB_COLORS);

     //ataşază hanble-ul la obținut la obiectul CBitmap
     ddBitmap.Attach(hDDBitmap);
```

```
    //selectează bitmap-ul in DC-ul aflat în memorie
    //astfel încât toate desenările să se aibă loc în imagine
    CBitmap* pTempBmp = dc.SelectObject(&ddBitmap);

    //începând de aici toate desenările se vor face pe imagine DDB

    //obține poziția curentă de scroll (din cauza barelor de scroll
    //coordonatele pot fi translatate)şi ajustează coordonatele
    CPoint pos = GetScrollPosition()+point;

    //creare a pen verde pentru desenare
    CPen pen(PS_SOLID, 1, RGB(0,255,0));

    //selectează pen-ul în DC
    CPen *pTempPen = dc.SelectObject(&pen);    //afişare text

    dc.TextOut(pos.x,pos.y, "test");
    dc.MoveTo(pos.x,pos.y); //afişare linie
    dc.LineTo(pos.x, pos.y-20);


    //selectează vechiul pen
    dc.SelectObject(pTempPen);
    //şi vechiul DDB
    dc.SelectObject(pTempBmp);

    //copiază pixelii din DDB în DIB-ul destinație
    GetDIBits(dc.m_hDC, ddBitmap, 0, dwHeight, lpDst, (LPBITMAPINFO)lpD,
    DIB_RGB_COLORS);

    END_PROCESSING("linie");
}
```

## 4.4. Activități practice

1. Pentru fiecare obiect individual dintr-o imagine etichetată calculați aria acestuia, centrul de masă, axa de alungire, perimetrul, factorul de subțiere şi elongația (factorul de aspect). Pentru afişarea valorilor calculate puteți opta pentru una dintre cele două variante prezentate în capitolul 4.3:
   a. afişarea într-un dialog a proprietăților geometrice pentru toate obiectele din imagine
   b. afişarea într-un MessageBox a proprietăților unui obiect atunci când se face double-click pe suprafața acestuia.
2. Afişați axa de alungire a obiectelor folosind funcțiile de desenare GDI.
3. Calculați şi afişați proiecțiile obiectelor.
4. Pentru fiecare obiect afişați peste imaginea destinație centrul de masă şi aria folosind funcțiile de desenare GDI.
5. **Salvați-vă ceea ce ați lucrat. Utilizați aceeaşi aplicație în laboratoarele viitoare. La sfârşitul laboratorului de procesare a imaginilor va trebui să prezentați propria aplicație cu algoritmii implementați!!!**

## Referințe

[1]. Umbaugh Scot E, *Computer Vision and Image Processing*, Prentice Hall, NJ, 1998, ISBN 0-13-264599-8