Lucrarea de laborator nr. 2

Operații de bază în prelucrarea imaginilor cu MATLAB

1. Obiectivele lucrării

Această lucrare de laborator conține două exemple pe care studenții trebuie să le studieze pentru disciplina **Prelucrarea Imaginilor**. Sunt exemple utile, care dau o primă idee asupra facilităților MATLAB şi asupra modului de folosire a pachetului *Image Processing Toolbox* (IPT).

Exemplul 1.1 se referă la câteva operații asupra imaginilor, operații incluse în IPT, cum sunt: citirea, scrierea şi afişarea imaginilor.

Exemplul 1.2 are în vedere operații mai complexe de prelucrare a imaginilor cum sunt: etichetarea regiunilor, măsurarea proprietăților regiunilor (obiectelor), operații aritmetice şi morfologice efectuate asupra imaginilor, modificarea contrastului imaginilor.

Pentru informații suplimentare asupra funcțiilor MATLAB este de dorit să se folosească meniul *Help* al mediului de programare.

După ce au fost parcurse cele două exemple, studenții vor efectua propriile teste asupra unor imagini din biblioteca individuală de imagini.

2. Desfăşurarea lucrării

Ce este IPT?

IPT este o colecție de M-funcții care extind posibilitățile mediului de programare MATLAB şi ajută la efectuarea unei game largi de operații de procesare a imaginilor:

- transformări spațiale
- operații de tip morfologic
- operații asupra vecinătăților unor pixeli
- filtrare liniară şi proiectarea filtrelor
- transformări
- analiza şi îmbunătățirea calității imaginilor
- înregistrarea imaginilor
- accentuarea imaginilor estompate
- operații asupra regiunilor de interes

Multe din funcțiile IPT sunt fişiere `*.m`, o succesiune de instrucțiuni MATLAB, care implementează *algoritmi* specifici pentru procesarea imaginilor. Codul MATLAB al acestor funcții poate fi accesat cu ajutorul instrucțiunii

```
type nume_funcție
```

Facilitățile oferite de IPT pot fi extinse prin elaborarea unor M-funcții proprii, sau prin folosirea unor combinații de funcții MATLAB din alte colecții de funcții MATLAB (*Signal Processing Toolbox*, *Wavelet Toolbox* etc.).

Exemplul 1.1

Acest exemplu are ca scop introducerea unor concepte de bază pentru procesarea imaginilor, inclusiv citirea şi scrierea imaginilor, egalizarea histogramei unei imagini şi obținerea unor informații despre imaginea curentă. Acest exemplu poate fi împărțit în următoarele etape:

Etapa 1: Citirea şi afişarea unei imagini
Etapa 2: Înțelegerea modului în care imaginea citită apare în spațiul de lucru MATLAB (*workspace*)
Etapa 3: Modificarea (îmbunătățirea) contrastului
Etapa 4: Scrierea unei imagini într-un fişier
Etapa 5: Obținerea informațiilor despre un fişier grafic

1. Citirea şi scrierea unei imagini

Este indicat să se șteargă spațiul de lucru MATLAB şi să se închidă toate ferestrele deschise pentru figuri:

```
clear, close all
```

Pentru a citi o imagine se foloseşte comanda `imread`. Exemplul care urmează citeşte una din imaginile conținute în *Biblioteca de Imagini* pe care utilizatorul a dezvoltat-o (sau una din imaginile incluse în IPT). Se citeşte imaginea `pout.tif` şi se memorează temporar într-un tablou numit I:

```
I = imread('pout.tif');
```

`imread` stabileşte că formatul grafic este *Tagged Image File Format* (TIFF). Pentru lista care include toate formatele grafice acceptate consultați documentația de referință `imread`.

În continuare, imaginea citită trebuie să fie afişată. IPT include două M-funcții pentru afişarea unei imagini: `imshow` şi `imtool`. `imshow` este funcția de bază; `imtool` declanşează *Image Tool* (mediu grafic integrat pentru afişarea imaginilor şi efectuarea unor operații simple de procesare). IT oferă toate posibilitățile de afişare ale `imshow` dar, în plus, oferă acces la mai multe unelte care pot fi folosite pentru navigarea în interiorul imaginilor sau explorarea imaginilor: *scroll bars*, *Pixel Region*, *Image Information*, *Contrast Adjustment*. Vom explora aceste facilități în ?????

Aşadar, pentru afişarea unei imagini se poate folosi oricare din cele două funcții amintite mai sus. În acest exemplu vom folosi `imshow`.

```
imshow(I)
```

2. Memorarea imaginii în spațiul de lucru MATLAB

Pentru a vedea modul în care funcția `imshow` memorează datele imagine în spațiul de lucru, se deschide fereastra *Workspace* în MATLAB. Sunt afişate aici informații despre toate variabilele care au fost create într-o sesiune de lucru MATLAB. Funcția `imread` a returnat datele imagine pentru variabila I, care este un tablou cu 291 x 240 elemente de tipul `uint8`. MATLAB poate memora imaginile sub forma unor tablouri `uint8`, `uint16` sau `double`.

De asemenea, se pot obține informații despre variabilele din spațiul de lucru prin comanda `whos`.

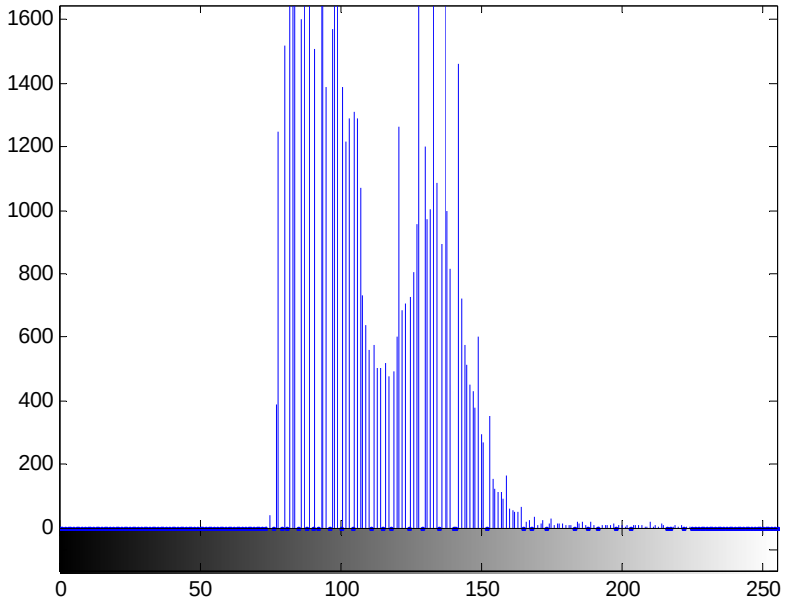
```
whos
Name      Size        Bytes    Class
  I       291x240     69840    uint8 array
Grand total is 69840 elements using 69840 bytes
```

3. Modificarea contrastului

După cum se poate observa, `pout.tif` este o imagine cu un contrast redus. Pentru a vedea distribuția intensității pixelilor în `pout.tif` se afişează histograma imaginii prin apelarea funcției `imhist`.

Observație. Apelarea funcției `imhist` trebuie să fie precedată de o comandă care să evite afişarea histogramei peste imaginea afişată anterior.

```
imhist(I)
```



Se poate observa că gama în care variază intensitatea pixelilor este destul de îngustă (majoritatea pixelilor au intensități între 76 şi 160). Nu este acoperită toată gama accesibilă [0, 255] şi lipsesc valorile maximă şi minimă care ar conduce la un contrast bun.

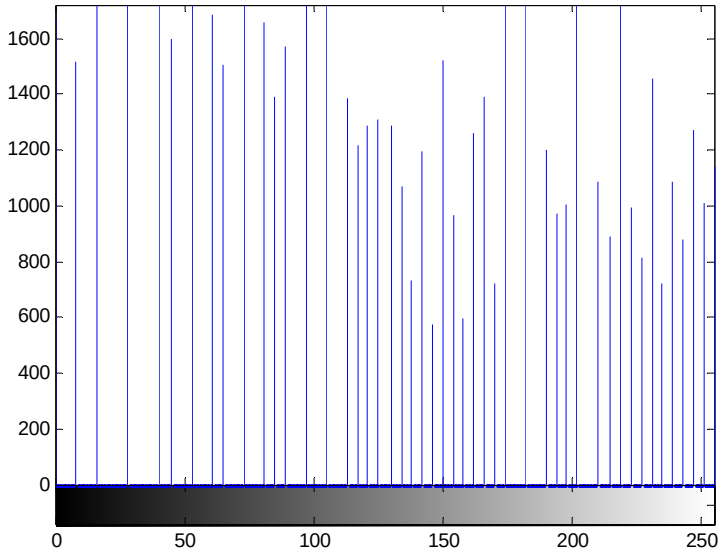
Sunt mai multe posibilități de a îmbunătăți contrastul unei imagini folosind IPT. O modalitate este aceea de a *egaliza* histograma, adică a repartiza valorile intensității peste toată gama posibilă; se realizează prin apelarea funcției `histeq`:

```
I2=histeq(I);
```

Apoi, se afişează noua imagine, I2, într-o nouă fereastră:

```
figure, imshow(I)
```

Se poate apela din nou `imhist` pentru a afişa histograma imaginii I2. Se observă „împrăştierea” valorilor intensității peste toată gama accesibilă de valori.



4. Scrierea unei imagini într-un fişier

Pentru a salva noua imagine modificată I2 sub forma unui fisier pe disc se foloseşte funcția `imwrite`. Dacă în comandă se include extensia `.png`, funcția `imwrite` va forma un fişier în format *Portable Network Graphics* (PNG); se pot specifica şi alte formate.

```
imwrite (I2, 'pout2.png');
```

5. Obținerea informațiilor despre un fişier grafic

Pentru a vedea ceea ce s-a obținut prin scrierea unui fişier pe disc se foloseşte funcția `iminfo`. Această funcție returnează informația despre imaginea respectivă: formatul ei, mărimea, dimensiunile etc.

```
imfinfo('pout2.png')
```

```
ans =

                Filename:  'pout2.png'
            FileModDate:  '25-Sep-2010 13:26:19'
               FileSize:  36938
                 Format:  'png'
          FormatVersion:  []
                  Width:  240
                 Height:  291
               BitDepth:  8
              ColorType:  'grayscale'
        FormatSignature:  [137 80 78 71 13 10 26 10]
               Colormap:  []
              Histogram:  []
          InterlaceType:  'none'
           Transparency:  'none'
  SimpleTransparencyData:  []
        BackgroundColor:  []
        RenderingIntent:  []
         Chromaticities:  []
                  Gamma:  []
             XResolution:  []
             YResolution:  []
          ResolutionUnit:  []
                 XOffset:  []
                 YOffset:  []
              OffsetUnit:  []
        SignificantBits:  []
           ImageModTime:  '25 Sep 2010 10:26:19 +0000'
                  Title:  []
                 Author:  []
            Description  []
              Copyright:  []
           CreationTime:  []
               Software:  []
             Disclaimer:  []
                Warning:  []
                 Source:  []
                Comment:  []
              OtherText:  []
```

Exemplul 1.2

Acest exemplu are ca scop introducerea unor noțiuni avansate pentru procesarea imaginilor. Sunt efectuate calcule statistice asupra obiectelor din imagini. Totuşi, înainte de a efectua aceste calcule, imaginea trebuie să fie „pre-procesată”, pentru ca rezultatele obținute să fie mai aproape de realitate. *Pre-procesarea* include şi formarea unui fond uniform în imaginea inițială şi convertirea acesteia într-o imagine binară. Acest exemplu poate fi împărțit în următoarele etape:

Etapa 1: Citirea şi afişarea unei imagini
Etapa 2: Estimarea valorii aproximative a pixelilor fondului
Etapa 3: Afişarea pixelilor fondului sub forma unei suprafețe
Etapa 4: Formarea unei imagini cu fond uniform
Etapa 5: Modificarea contrastului unei imagini uniforme
Etapa 6: Crearea versiunii binare a unei imagini
Etapa 7: Determinarea numărului de obiecte dintr-o imagine
Etapa 8: Examinarea matricii de etichete
Etapa 9: Afişarea matricii de etichete sub forma unei imagini pseudo-color
Etapa 10: Măsurarea proprietăților obiectelor dintr-o imagine
Etapa 11: Efectuarea unor calcule statistice asupra obiectelor dintr-o imagine

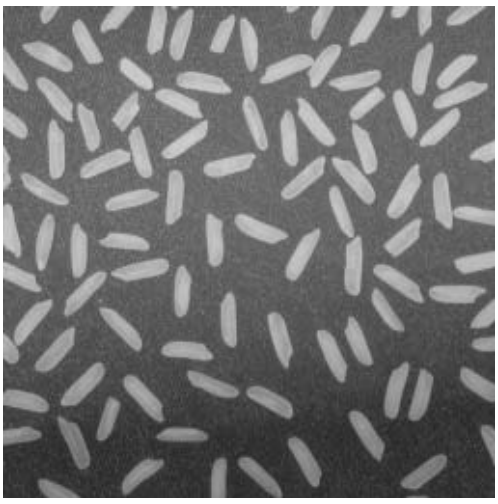
1. Citirea şi afişarea unei imagini

Ştergeți toate variabilele din spațiul de lucru MATLAB, închideți toate figurile deschise şi închideți toate accesoriile deschise (*Image Tools*):

```
clear, close all, imtool close all
```

Citiți şi afişați imaginea `rice.png`.

```
I = imread('rice.png');
imshow(I)
```



2. Estimarea valorii aproximative a pixelilor fondului

În imaginea anterioară, fondul este iluminat mai intens în centrul imaginii şi mai puțin intens în partea de jos. În această etapă, vom folosi operația morfologică numită *deschidere* pentru a estima iluminarea fondului. O deschidere este alcătuită dintr-o operație de *erodare* urmată de o operație de *dilatare*, folosind acelaşi element structural pentru ambele operații. Deschiderea morfologică are ca efect înlăturarea obiectelor care nu pot conține în întregime elementul structural.

Vom folosi funcția `imopen` pentru a efectua operația de deschidere morfologică. Totuşi, înainte de acest apel, trebuie să se formeze un element structural: acesta va fi un disc a cărui rază este de 15 pixeli şi va fi creat cu funcția `strel`. Pentru a elimina boabele de orez din imaginea inițială elementul structural trebuie să fie suficient de mare pentru a nu putea fi inclus în întregime într-un bob de orez.

```
background = imopen(I,strel('disk',15));
```

Pentru a vizualiza fondul:

```
figure, imshow(background)
```

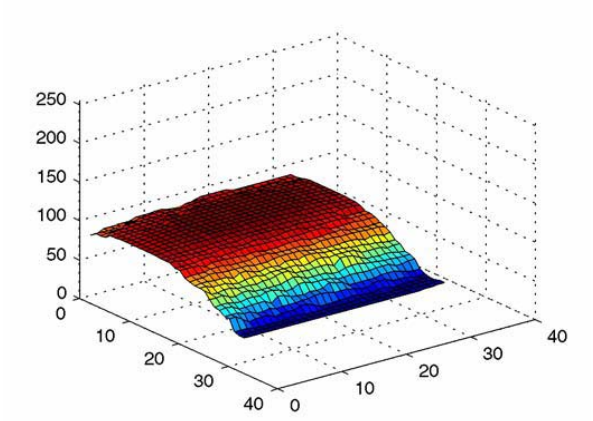
3. Afişarea pixelilor fondului sub forma unei suprafețe

Pentru a obține o reprezentare grafică a intensității pixelilor fondului unei imagini sub forma unei suprafețe se foloseşte funcția `surf`. `surf` crează suprafețele parametrice care permit reprezentarea grafică a funcțiilor într-o regiune dreptunghiulară. Această funcție se aplică clasei de date `double`; înainte de aplicare este nevoie să se convertească `background` cu ajutorul comenzii `double`.

```
    figure,surf(double(background(1:8:end,1:8:end))),zlim
([0 255]);
    set(gca,'ydir','reverse');
```

Acest exemplu foloseşte sintaxa MATLAB care permite afişarea unui singur pixel din 8 în fiecare direcție; altfel, suprafața afişată ar fi fost prea densă. De asemenea, este setată scala graficului pentru a se potrivi mai bine cu gama datelor `uint8` şi se inversează axa y pentru a oferi o observare mai bună a datelor (pixelii din partea de jos a imaginii apar în partea din față a graficului suprafeței).

În graficul afişat, [0, 0] reprezintă originea, sau colțul din stânga sus al imaginii. Partea cea mai înaltă a curbei indică faptul că valorile cele mai mari ale pixelilor din `background` (şi cele din `rice.png`) apar în apropierea liniilor din mijloc ale imaginii. Valorile cele mai mici ale pixelilor apar în partea de jos a imaginii şi sunt reprezentate în grafic de partea cea mai de jos a curbei.



4. Formarea unei imagini cu fond uniform

Pentru a forma un fond mai uniform al unei imagini se scade fondul, `background` în acest exemplu, din imaginea inițială I.

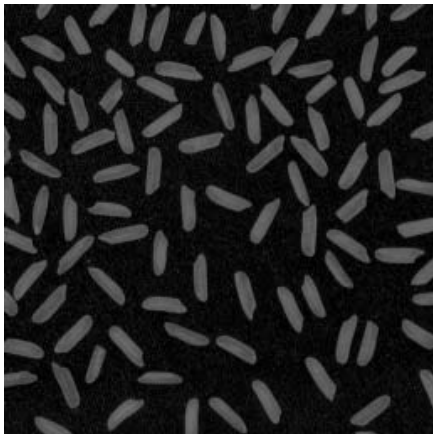
```
I2 = imsubtract(I, bacground);
```

Deoarece operația de scădere, ca multe alte operații în MATLAB, se aplică numai clasei de date `double`, trebuie să se folosească funcția `imsubtract` din IPT.

Afişarea noii imagini se obține prin

```
figure, imshow(I2)
```

Se poate observa fondul uniform în comparație cu cel inițial.



5. Modificarea contrastului unei imagini uniforme

După efectuarea operației de scădere imaginea are un fond uniform dar este prea întunecată. Pentru modificarea contrastului se poate folosi funcția imadjust.

```
I3 = imadjust(I2);
```

Această funcție creşte contrastul imaginii prin saturarea cu 1% a valorilor pixelilor din I2 atât la valoarea minimă cât şi la valoarea maximă şi prin comprimarea valorilor intensității astfel încât să corespundă cu gama `uint8`.
Noua imagine se afişează.

```
figure, imshow(I3);
```

6. Crearea versiunii binare a unei imagini

Versiunea binară a unei imagini date se obține prin operația numită detecție de prag. Funcția `graythresh` calculează automat o valoare potrivită a pragului care va fi folosit pentru conversia unei imagini cu niveluri multiple de gri într-o imagine binară. Conversia propriu-zisă se realizează cu funcția `im2bw`.

```
level = graythresh(I3);
bw = im2bw(I3,level);
figure, imshow(bw)
```

Imaginea binară returnată de `im2bw` (numită `bw` în acest exemplu) este din clasa `logical`, după cum se poate observa la apelul `whos`.

```
whos
```

MATLAB va răspunde cu

```
Name         Size       Bytes    Class
I            256x256    65536    uint8 array
I2           256x256    65536    uint8 array
I3           256x256    65536    uint8 array
background   256x256    65536    uint8 array
bw           256x256    65536    logical array
level        1x1        8        double array

Grand total is 327681 elements using 327688 bytes
```

7. Determinarea numărului de obiecte dintr-o imagine

După conversia imaginii într-o imagine binară, utilizatorul poate folosi funcția `bwlabel` pentru a determina numărul de obiecte din imagine (în acest exemplu, numărul de boabe de orez). Funcția `bwlabel` etichetează toate componentele din imaginea binară (numită, în acest exemplu, `bw`) şi returnează numărul componentelor găsite în imagine prin valoarea de ieşire, `numObjects`.

```
[labeled, numObjects] = bwlabel(bw, 4);

numObjects

ans =

  101
```

Precizia rezultatului depinde de:
- dimensiunea obiectelor din imagine
- gradul de suprapunere a unor obiecte (dacă obiectele se suprapun, ele pot fi etichetate ca fiind un singur obiect)
- precizia de aproximare a fondului
- tipul de conectivitate selectat. Parametrul 4, transmis funcției bwlabel însemană că pixelii sunt tetraconectați.

8. Examinarea matricii de etichete

Pentru a putea înțelege mai bine semnificația matricii de etichete care este returnată de funcția `bwlabel`, vom explora în această etapă valorile pixelilor imaginii. Sunt mai multe modalități pentru a realiza această operație. De exemplu, se poate folosi funcția `imcrop` pentru a selecta o zonă mică din imagine. Sau, se poate folosi opțiunea *Pixel Region* din IPT pentru a examina valorile pixelilor. Se afişează matricea de etichete (numită `labeled`, în acest exemplu) cu `imshow`

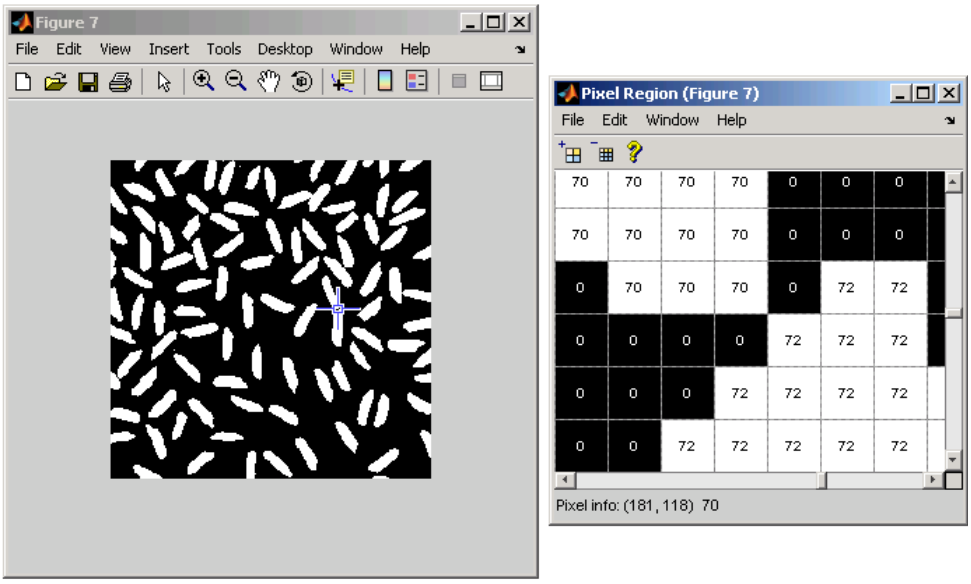
```
figure, imshow(labeled);
```

Se porneşte opțiunea *Pixel Region*

```
impixelregion
```

În mod implicit, se asociază cu imaginea din figura curentă. *Pixel Region* desenează un dreptunghi, numit *dreptunghiul pixelilor regiunii*, în centrul părții vizibile a imaginii. Acest dreptunghi definește pixelii care vor fi afișați în *Pixel Region*. Pe măsură ce deplasați dreptunghiul, *Pixel Region* actualizează valorile afișate în fereastră.

Următoarele figuri prezintă *Image Viewer* cu dreptunghiul poziționat peste două boabe de orez. Se poate observa că pixelii care aparțin boabelor de orez au valorile asignate prin funcția `bwlabel`, iar pixelii fondului au valoarea 0.

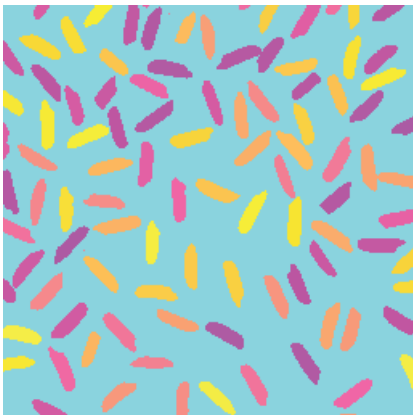


9. Afişarea matricii de etichete sub forma unei imagini pseudo-color

O metodă bună pentru a observa matricea de etichete este afişarea acesteia sub forma unei imagini indexate *pseudo-color*. În imaginea pseudo-color, numărul care identifică fiecare obiect în matricea de etichete este transformat într-o culoare. Culorile din imagine fac mai uşoară diferențierea obiectelor.

Pentru a afişa matricea de etichete în acest mod, se foloseşte funcția `label2rgb` şi specificând toți parametrii ceruți de aceasta (culoarea fondului, paleta de culori şi modul în care obiectelor din matricea de etichete li se atribuie culori din paleta respectivă).

```
pseudo_color = label2rgb(labeled, @spring, 'c',
'shuffle');
imshow(pseudo_color);
```



10. Măsurarea proprietăților obiectelor dintr-o imagine

Funcția `regionprops` permite măsurarea proprietăților regiunilor (obiectelor) dintr-o imagine şi returnează valorile sub forma unui tablou. Atunci când se aplică unei imagini ale cărei componente au fost etichetate, ea conduce la crearea unui element de structură pentru fiecare componentă din imagine.

Acest exemplu foloseşte `regionprops` pentru a forma un tablou structural care conține unele proprietăți de bază ale imaginii numite `labeled`. Atunci când se setează parametrul properties la `'basic'`, funcția `regionprops` va returna rezultatele a trei măsurări de bază: aria, centrul de masă (*centroid*) şi dreptunghiul circumscris (*bounding box*). Acesta din urmă este cel mai mic dreptunghi care conține o regiune din imagine sau, în exemplul nostru, o boabă de orez.

```
graindata = regionprops(labeled, 'basic')
```

MATLAB va răspunde cu

```
graindata =

101x1 struct array with fields:
    Area
    Centroid
    BoundingBox
```

În continuare, pentru a afla aria componentei etichetate cu 51, trebuie să se acceseze elementul 51 din câmpul `Area` din tabloul structural `graindata`. Atenție, numele câmpurilor structurii fac diferența între majuscule şi litere mici.

```
graindata(51).Area
```

Rezultatul returnat este

```
ans =
  140
```

Pentru a găsi cel mai mic dreptunghi circumscris şi centrul de masă pentru aceeaşi componentă, se foloseşte

```
graindata(51).BoundingBox, graindata(51).Centroid

ans =
     107.5000   4.5000  13.0000   20.0000

ans =
     114.5000   15.4500
```

11. Efectuarea unor calcule statistice asupra obiectelor dintr-o imagine

În continuare, vom folosi funcții MATLAB pentru a calcula unele proprietăți statistice ale obiectelor. Mai întâi, putem folosi funcția `max` pentru a determina dimensiunea celei mai mari regiuni (a celui mai mare bob de orez). În exemplul nostru, cel mai mare bob de orez este acela obținut prin atingerea a două boabe.

```
max([graindata.Area])
```

Răspunsul va fi

```
ans =
     404
```

Se poate folosi comanda `find` pentru a obține eticheta componentei (bobul de orez) care are această arie.

```
biggrain = find([graindata.Area]==404)
```

Răspunsul va fi

```
biggrain =
     59
```

Vom putea găsi media ariilor tuturor regiunilor din imagine (aria medie a unui bob de orez):

```
mean([graindata.Area])
```

Vom obține

```
ans =
     175.0396
```

În sfârşit, putem obține histograma care conține 20 grupuri pentru a observa distribuția dimensiunilor boabelor de orez. Această histogramă arată că cele mai multe boabe de orez au dimensiuni între 150 şi 250 pixeli.

```
hist([graindata.Area], 20)
```

3. Conținutul referatului de laborator

Referatul lucrării trebuie să fie prezentat în modul următor, pe CD:

a. O *Bibliotecă de Imagini* individuală, originală, cu imagini (diferite formate şi dimensiuni) asupra cărora vor fi aplicate funcțiile MATLAB.

b. Un *repertoriu*, organizat sub forma unui tabel, al funcțiilor *învățate* şi *testate* în această lucrare (inclusiv caracteristicile principale ale acestor funcții).

c. Fişiere M (grupuri de instrucțiuni MATLAB) care demonstrează aplicarea funcțiilor din această lucrare *pe imagini particulare* din Biblioteca de Imagini. Rezultatele aplicării funcțiilor pe imagini particulare.

d. Modul de folosire a CD-ului.

e. *Răspunsuri* la probleme şi/sau întrebări.

f. *Observații*, propuneri de dezvoltare sau alte elemente care demonstrează aprofundarea noțiunilor din lucrare.

g. Dovezi scrise care demonstrează parcurgerea şi analiza unor programelor „*demos*" din IPT.

Întrebări şi probleme

a. Ce alte formate de imagini mai pot fi salvate în IPT?

b. Ce alte informații despre imagine se afişează cu `iminfo`?

c. Pentru toate funcțiile studiate în această lucrare, completați modul de folosire cu explicații legate de parametrii care apar în forma generală de apelare. Dați exemple şi prezentați pe CD modul în care parametrii funcției afectează rezultatul obținut prin apelare.

d. Aplicați noțiunile învățate pe o imagine care conține mai multe monede împrăştiate pe o suprafață plană. Scrieți un program MATLAB prin care să se determine suma conținută în imaginea respectivă.