確認テストの解答

｛其の一｝

【問題】演習②の1細胞をクロップしたステップ以降をマクロとして記述せよ．
縁の幅は6ピクセルとせよ．

【答え】

以下のようなマクロになる．

```
title = getTitle();
c1 = "C1-" + title;
c2 = "C2-" + title;
run("Split Channels");
selectWindow(c1);
c1ID = getImageID();
selectWindow(c2);
c2ID = getImageID();

run("Gaussian Blur...", "sigma=1");
run("Auto Threshold", "method=Li white");
run("Open");
run("Fill Holes");
run("Duplicate...", "title=C2-2-" + title);
c2bID = getImageID();
setOption("BlackBackground", true);
for (i = 0; i < 3; i++)
    run("Erode");
selectImage(c2ID);
for (i = 0; i < 3; i++)
    run("Dilate");
```

```
22    imageCalculator("Subtract create", c2ID, c2bID);
23    rimID = getImageID();
24    run("Create Selection");
25    run("Make Inverse");
26    selectImage(c1ID);
27    run("Restore Selection");
28    run("Set Measurements...", "area mean min integrated redirect=None decimal=2");
29    run("Measure");
30
31    //cleanup
32    selectImage(c2ID); close();
33    selectImage(c2bID); close();
34    selectImage(rimID); close();
```

1〜8行目はこれまでにも何度か登場したように，2チャネルの画像を分割し，それぞれのチャネルの画像のImageIDを取得している．10〜13行目までが核の二値化の工程である．分節化された画像の膨張処理と侵食処理は17〜21行目までである．このようにforループを使えば，処理の回数を調節するのに便利である．ここではループが3回なので，22行目で行われる引き算の結果は6ピクセルの核の縁を中心とする帯状の構造の分節化となる．24・25行目が二値画像を選択領域に変換するコマンドである．26〜29行目がその選択領域を使った測定である．31行目以降は，このマクロで生成される画像の数が多いので，不要なものを閉じている．

26〜29行目の選択領域を使った測定の部分は，ROI managerを使うと以下のようになる．

```
ROI manager("Add");
selectImage(c1ID);
run("Set Measurements...", "area mean min integrated redirect=None decimal=2");
ROI manager("select", 0)
ROI manager("Measure");
```

ROI managerをマクロから使う場合，基本的にはROI managerのウィンドウにあるボタンの名前を引数にすれば，ボタンをクリックすることと同じ操作ができる．上のスニペットでは1行目で選択領域をROI managerに登録し，それを核膜タンパク質のチャネルでアクティブにする．この際にROI manager(**"select", index**)のコマンドを使う．indexはROI managerにリストされた選択領域のインデックスということで，今回は1つしかないので最初のインデックスである0になる．測定の実行はrun(**"Measure"**)でもよいのだが，ROI managerを使っているのでROI managerに付属している測定機能を使った．このコマンドからわかるように，ROI managerを使えば複数の選択領域を登録して任意の領域を自由に測定することが可能になる．

また，今回のようにマクロが生成する画像が多い場合，実行中にいろいろなウィンドウが現れてチカチカする．このようなときには**“setBatchMode”**を使って描画を一時的に停止するとよい．一番最初の行に**“setBatchMode(true)”**を挿入し，最初から描画を一時停止する．また一番最後の行でこの一時停止を解除して，結果の画像を描画する．

以上の改造を行ったのが以下のコードである．

```
1     setBatchMode(true);
2     title = getTitle();
3     c1 = "C1-" + title;
4     c2 = "C2-" + title;
5     run("Split Channels");
6     selectWindow(c1);
7     c1ID = getImageID();
8     selectWindow(c2);
9     c2ID = getImageID();
10
11    run("Gaussian Blur...", "sigma=1");
12    run("Auto Threshold", "method=Li white");
13    run("Open");
14    run("Fill Holes");
15    run("Duplicate...", "title=C2-2-" + title);
16    c2bID = getImageID();
17    setOption("BlackBackground", true);
```

```
18      for (i = 0; i < 3; i++)
19          run("Erode");
20      selectImage(c2ID);
21      for (i = 0; i < 3; i++)
22          run("Dilate");
23      imageCalculator("Subtract create", c2ID, c2bID);
24      rimID = getImageID();
25      run("Create Selection");
26      run("Make Inverse");
27      ROI manager("Add");
28      selectImage(c1ID)
29      run("Set Measurements...",
30          "area mean min integrated redirect=None decimal=2");
31      ROI manager("select", 0);
32      ROI manager("Measure");
33
34      //clean up
35      selectImage(c2ID); close();
36      selectImage(c2bID); close();
37      selectImage(rimID); close();
38      setBatchMode("exit and display");
```

改造前と比較すると，実行速度が格段に速くなっていることがわかるだろう．最初のマクロの実行にかかる時間のほとんどはじつはディスプレイへの描画にかかっている時間なのである．**“setBatchMode”**によって，このムダな時間を省くことができるとともに，処理中の表示もスマートになる．

なお，このコードは以下のGithubのリンクにも掲載した．
https://gist.github.com/cmci/9845061
改造前と改造後を比較したいときには，差分を見るとわかりやすい．
https://gist.github.com/cmci/9845061/revisions

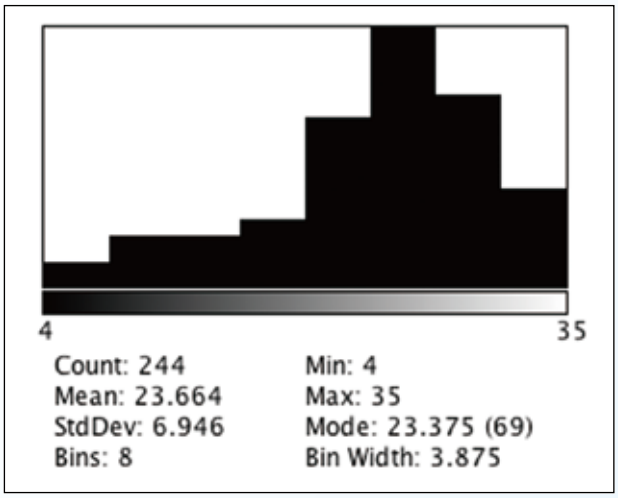
｛ 其の二 ｝

【問題】サンプル画像[quantumdots.tif]を開き，ドットの数を数えよ．ドットが重なって分節化されないように注意せよ．ドットの大きさ（面積）の分布をヒストグラムで示せ．

【 答え 】

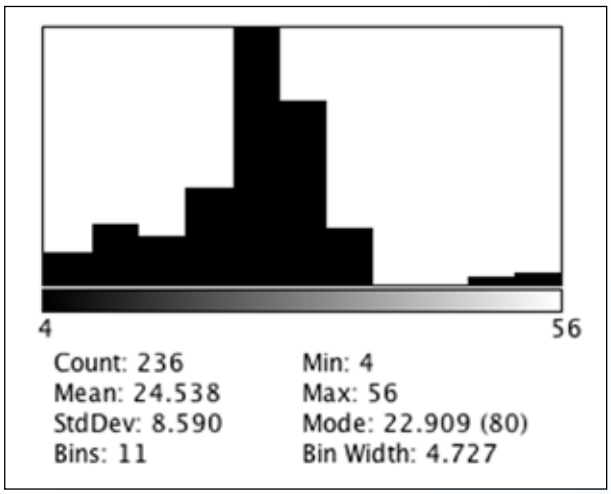
画像の背景をよく見るとわかるのだが，均一な背景ではなく濃淡がある．そこでまず背景を差し引く．今回は**[Subtract Backgound...]**によって行う．**Rolling ball radius**を5に設定し，チェックボックスはすべて選択しない．**OK**をクリックして実行すると，背景は完全に均一ではないが，元画像よりもかなり改善されていることがわかるだろう．次に**[Auto Local Threshold]**でMethod **Bernsen**を選択（最初にTry Allを選んで結果を比較して適切なアルゴリズムを選ぶ），Diameterを30として画像を二値化する．重なっているドットを分離するため**[Watershed]**を処理する．最後に測定である．**[Set measurements...]**で**“Area”**が選択されていることを確認した後，**[Analyze particles...]**を実行する．**Size**を3-Infinityとし，**Show**は**“Outlines”**，チェックボックスは**“Display results”**，**“Clear results”**，**“Exclude on edges”**，**“Include holes”**を選択する．

私の結果では244のq-dotsが分節化され，面積のヒストグラムはResults tableに独自についている**[Results > Distribution...]**によってヒストグラムをプロットした（図1）．



■図1　q-dotの面積のヒストグラム

なお，**[Watershed]**を行わない場合は236個が分節化される．面積のヒストグラムを見ると図2のようである．
右の方に小さなピークがある．かなり大きなドットあることがこのことからわかる．これらのドットの面積がモードの値である22の2倍程度であることから，これらは2つのq-dotsが1つのq-dotとして分節化されたものであると推測される．上のヒストグラムと比べれば分水嶺変換がこれらの重なっているドットを分割したことがわかる．



■図2　分水嶺変換をしない場合のqdotの面積のヒストグラム