**Computer and Robot Vision**

**Homework#2**

R01944040 柳成蔭

這次的作業是基本的圖像處理，(a) 取threshold=128來將lena圖變成binary image，(b)得到lena圖的histogram，(c)得到binary image 中的connected components。

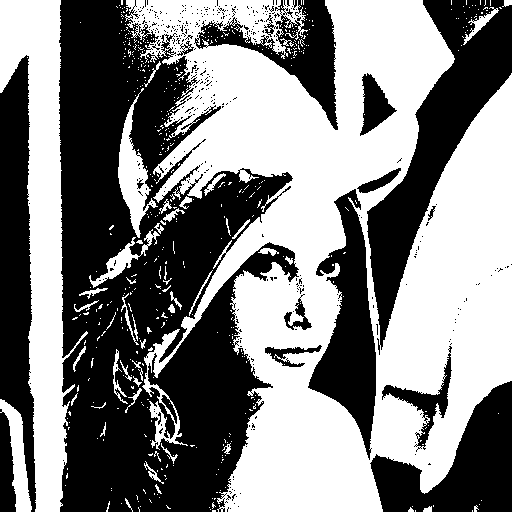
我使用VS2012編寫程式

1. binary image

對圖中的每個pixel進行判斷，如果值小於等於127，就設為0；如果大於127，就設為255。

|  |
| --- |
| for(i=0;i<=imgBinary.rows-1;i++)  {  for(j=0;j<=imgBinary.cols-1;j++)  {  if(imgBinary.at<uchar>(i,j)<=127)  imgBinary.at<uchar>(i,j)=0;  else  imgBinary.at<uchar>(i,j)=255;  }  } |

經過設threshold之後的結果：



1. histogram

統計每個值對應的pixel的個數，記錄在histogram[]中。

|  |
| --- |
| Mat\_<uchar>::iterator iteK=img.begin<uchar>(),  iteKEnd=img.end<uchar>();  while (iteK!=iteKEnd)  {  histogram[(\*iteK)]++;  iteK++;  } |

畫出histogram：



1. connected components

我使用4-connected neighborhood detection。

第一遍掃描將不是0的點進行label。如果是左、上都為0的點，直接新增一個label到linked中。

|  |
| --- |
| if( (i==0 && j==0)  || ( i==0 && label[i][j-1]==0 )  || ( label[i-1][j]==0 && j==0 )  || ( label[i-1][j]==0 && label[i][j-1]==0 ) ) //如果上和左是空的  {  //把labelNumber加到linked[labelNumber]  newNode.root=0;  newNode.nodeLabel=labelNumber;  newNode.nodeCount=1;    newNode.top=i;  newNode.down=i;  newNode.left=j;  newNode.right=j;  linked.push\_back(newNode);  label[i][j]=labelNumber;  labelNumber++;  } |

如果是左、上已經有label的點，就要找到左、上各自的root，比較之後歸到同一個set。

|  |
| --- |
| else  {  if(label[i-1][j]==0)  {  label[i][j]=label[i][j-1];  linked[ label[i][j] ].nodeCount++;  }  else if(label[i][j-1]==0)  {  label[i][j]=label[i-1][j];  linked[ label[i][j] ].nodeCount++;  }  else  {  label[i][j]=label[i-1][j];  linked[ label[i][j] ].nodeCount++;  int p=label[i][j-1];  int q=label[i-1][j];    while(linked[p].root>0)  {  p=linked[p].root;  }  while(linked[q].root>0)  {  q=linked[q].root;  }    if(linked[p].nodeLabel > linked[q].nodeLabel)  linked[q].root=p;  else if(linked[p].nodeLabel < linked[q].nodeLabel)  linked[p].root=q;  else  {  linked[p].root=0;  linked[q].root=0;  }  }  if(i<linked[ label[i][j] ].top)  linked[ label[i][j] ].top=i;  if(i>linked[ label[i][j] ].down)  linked[ label[i][j] ].down=i;  if(j<linked[ label[i][j] ].left)  linked[ label[i][j] ].left=j;  if(j>linked[ label[i][j] ].right)  linked[ label[i][j] ].right=j;  }  } |

之後將屬於同一個set的所有label的count數加總起來，并確定該set中上下左右位置值最大的點。如果pixel的數量大於500，就輸出pixel的數量，并把連通的區域框出來：

|  |
| --- |
| size\_t sizeLinked=linked.size();  for (int k = 1; k < sizeLinked ; k++)  {  if(linked[k].root!=0)  {  linked[linked[k].root].nodeCount=linked[linked[k].root].nodeCount + linked[k].nodeCount;    if(linked[k].top<linked[linked[k].root].top)  linked[linked[k].root].top=linked[k].top;  if(linked[k].down>linked[linked[k].root].down)  linked[linked[k].root].down=linked[k].down;  if(linked[k].left<linked[linked[k].root].left)  linked[linked[k].root].left=linked[k].left;  if(linked[k].right>linked[linked[k].root].right)  linked[linked[k].root].right=linked[k].right;  }  }  cvtColor(imgLabel,imgConCom,CV\_GRAY2RGB);  for (int k = 1; k < sizeLinked ; k++)  {  if(linked[k].root==0 && linked[k].nodeCount>=500)  {  cout << linked[k].nodeCount << endl;  Point a,b,c;  a.x=linked[k].left;  a.y=linked[k].top;  b.x=linked[k].right;  b.y=linked[k].down;  c=a+b;  c.x=c.x/2;  c.y=c.y/2;  circle( imgConCom,c,6,CV\_RGB(0,255,0),2 );  rectangle(imgConCom,a,b,CV\_RGB(255,120,120),2);  }  } |

找出4-connected neighborhood的connected components後的結果：

