**Computer and Robot Vision**

**Homework#7**

R01944040 柳成蔭

這次的作業是對原圖進行down sample，然後進行Thinning操作。

我使用VS2012編寫程式

先將binary的Lena圖從512x512 Downsample到64x64: 用8x8的 block作為一個unit, 選左上的pixel作為新64x64圖的pixel值。

|  |
| --- |
| //downsample  Mat imgDownSample(66,66,CV\_8UC1,Scalar(0)); //the boundary is zero  for(int i=1; i<=imgDownSample.rows-2; i++)  {  for(int j=1; j<=imgDownSample.cols-2; j++)  { imgDownSample.at<uchar>(i,j)=imgBinary.at<uchar>(8\*(i-1),8\*(j-1));  }  } |

Down sample result：



然後thinning operator分為三個步驟：

（1）Mark-Interior/Border-Pixel

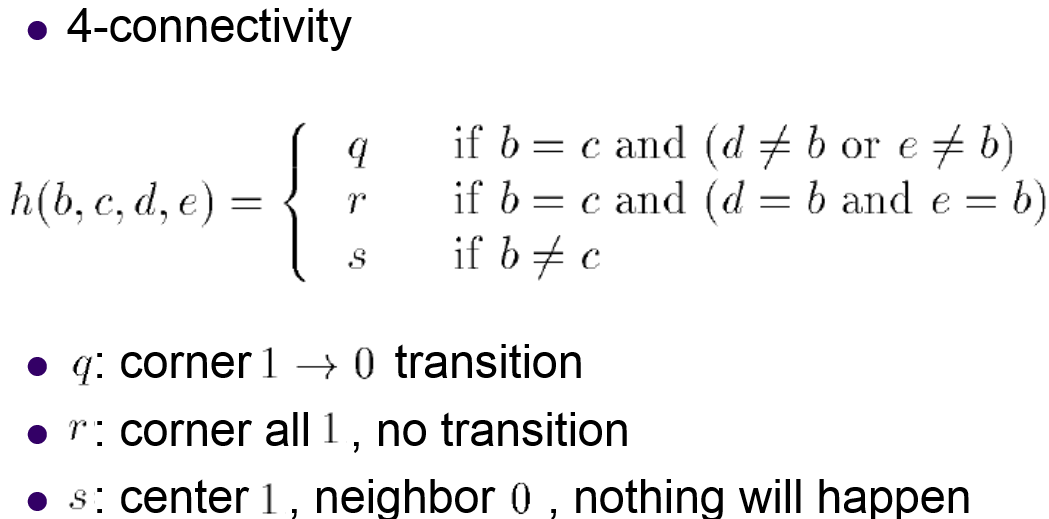
（2）The pair relationship

（3）Connected shrink

循環做直到shrink后的結果不會變化。

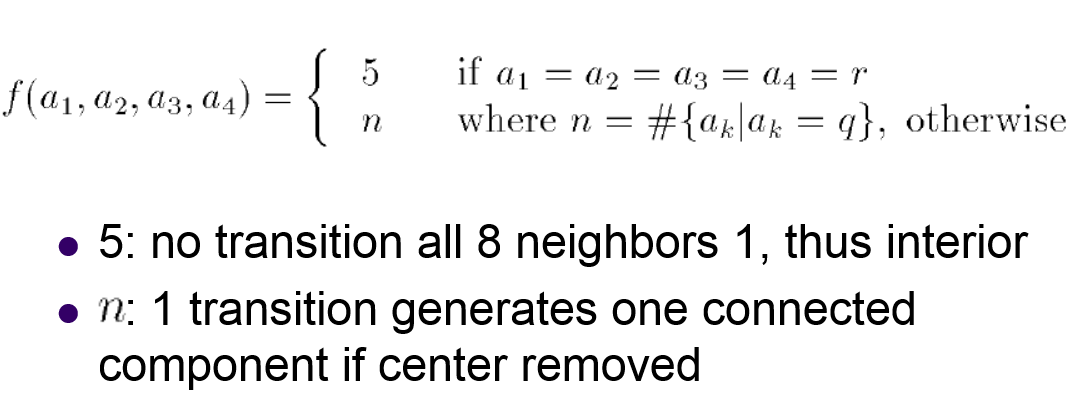
1. Yokoi Connectivity Number

Formula h



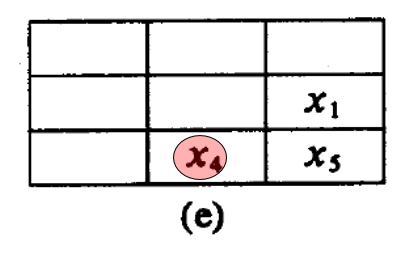
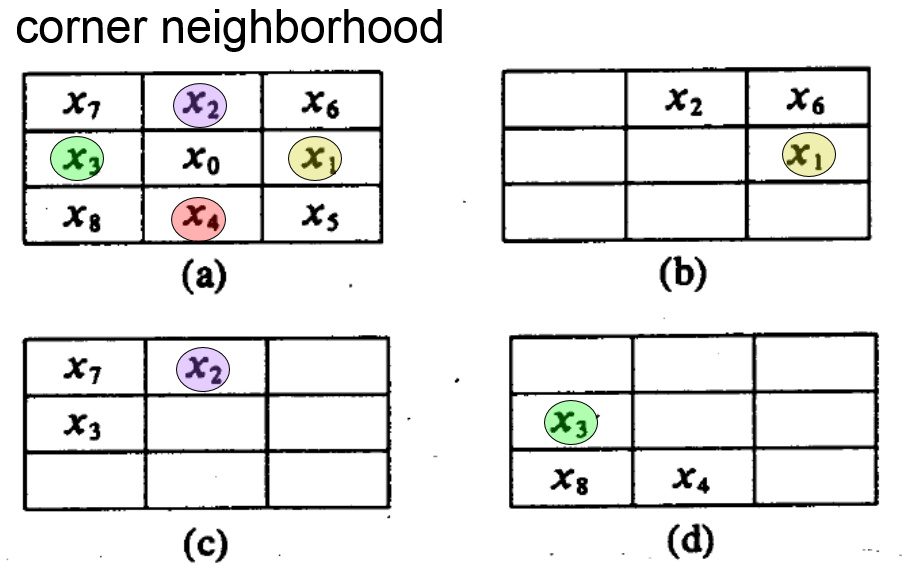
|  |
| --- |
| int h(int b, int c, int d, int e)  {  if( b==c && (d!=b || e!=b) )  return q;  else if( b==c && (d==b && c==b) )  return r;  else if(b!=c)  return s;  else  return -1;  } |

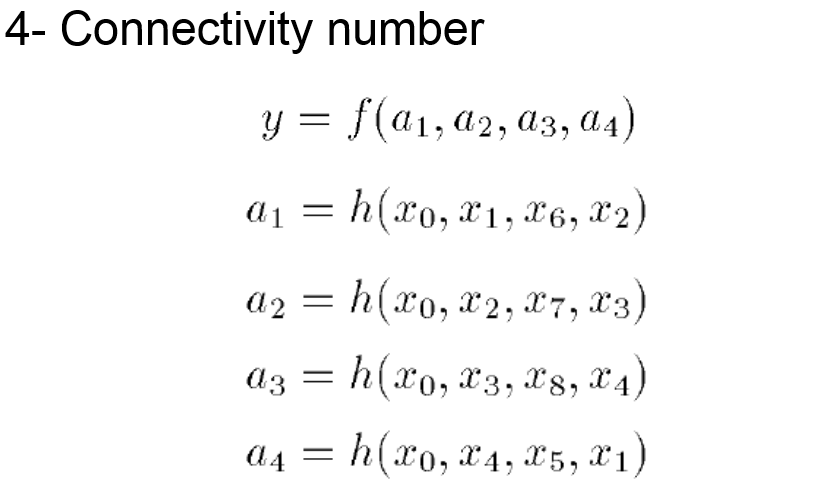
Formula f



|  |
| --- |
| int f(int a1, int a2, int a3, int a4)  {  if(a1==r && a2==r && a3==r && a4==r)  return 5;  else  {  int n=0;  if(a1==q)  n++;  if(a2==q)  n++;  if(a3==q)  n++;  if(a4==q)  n++;  return n;  }  } |

Yokoi number



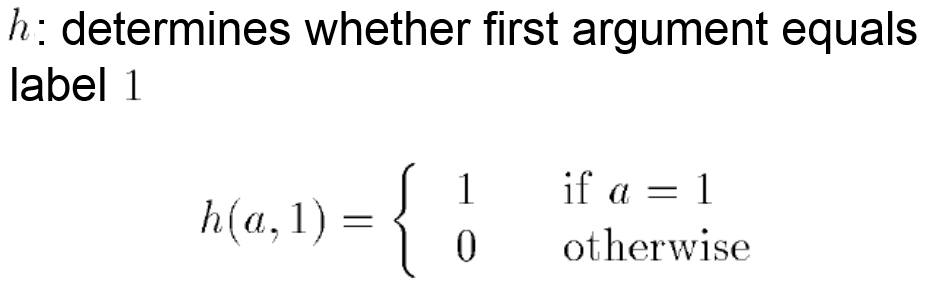


用Formula h來統計四個corner的連通值，用Formula f判斷中間點的Yokoi number。

|  |
| --- |
| void Yokoi(Mat src, Mat res)  {  for(int i=1; i<=src.rows-2; i++)  {  for(int j=1; j<=src.cols-2; j++)  {  if(src.at<uchar>(i,j)==255)  {  int x[9];  x[0]=src.at<uchar>(i,j);  x[1]=src.at<uchar>(i,j+1);  x[2]=src.at<uchar>(i-1,j);  x[3]=src.at<uchar>(i,j-1);  x[4]=src.at<uchar>(i+1,j);  x[5]=src.at<uchar>(i+1,j+1);  x[6]=src.at<uchar>(i-1,j+1);  x[7]=src.at<uchar>(i-1,j-1);  x[8]=src.at<uchar>(i+1,j-1);  int a1=h(x[0],x[1],x[6],x[2]);  int a2=h(x[0],x[2],x[7],x[3]);  int a3=h(x[0],x[3],x[8],x[4]);  int a4=h(x[0],x[4],x[5],x[1]);  res.at<uchar>(i,j)=f(a1,a2,a3,a4);  }  else  {  res.at<uchar>(i,j)=6;  }  }  }  } |

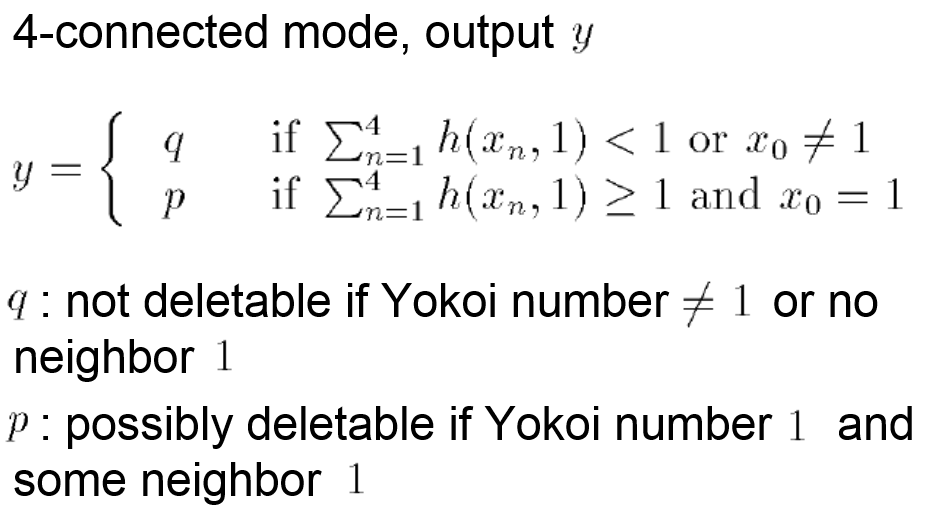
1. Pair Relationship

Formula hp



|  |
| --- |
| int hp(int a)  {  if(a==1)  return 1;  else  return 0;  } |

Formula yp



|  |
| --- |
| int yp(int x0, int x1, int x2, int x3, int x4)  {  int hpSum=hp(x1)+hp(x2)+hp(x3)+hp(x4);  if(hpSum<1 || x0!=1)  return q;  else if(hpSum>=1 && x0==1)  return p;  else  return -1;  } |

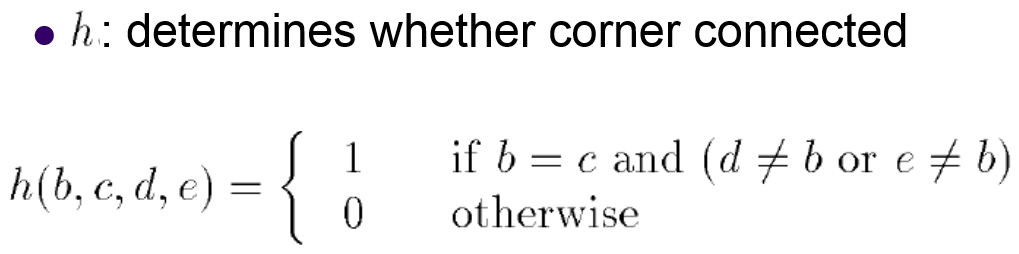
Pair relationship

對Yokoi的結果求出Pair relationship的對照表。

|  |
| --- |
| void Pair(Mat src, Mat res)  {  for(int i=1; i<=src.rows-2; i++)  {  for(int j=1; j<=src.cols-2; j++)  {  int x0=src.at<uchar>(i,j);  int x1=src.at<uchar>(i,j+1);  int x2=src.at<uchar>(i-1,j);  int x3=src.at<uchar>(i,j-1);  int x4=src.at<uchar>(i+1,j);  res.at<uchar>(i,j)=yp(x0,x1,x2,x3,x4);  }  }  } |

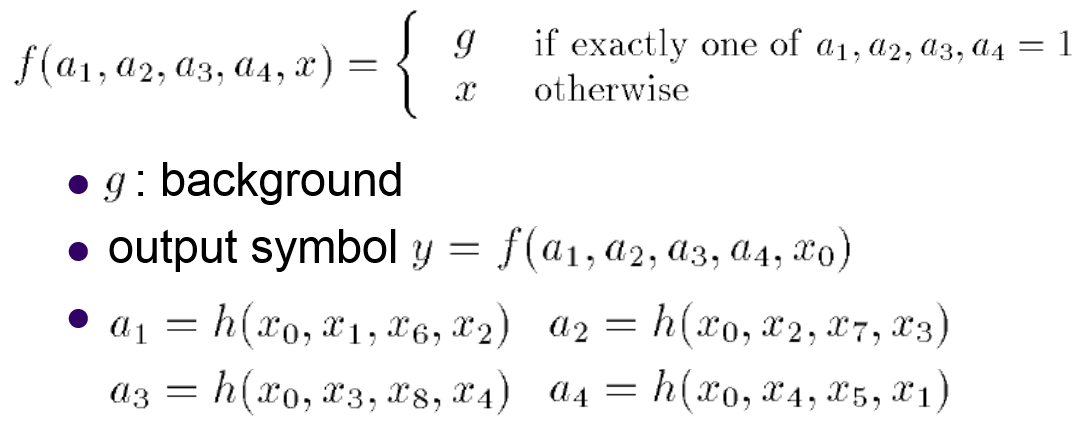
1. Connected Shrink Operator

Formula hs



|  |
| --- |
| int hs(int b, int c, int d, int e)  {  if( b==c && (d!=b || e!=b) )  return 1;  else  return 0;  } |

Formula fs



|  |
| --- |
| int fs(int a1, int a2, int a3, int a4, int x)  {  if((a1+a2+a3+a4)==1)  return g;  else  return x;  } |

Shrink operator

對down sample的結果，根據pair relationship進行shrink。

|  |
| --- |
| void Shrink(Mat src, Mat P, Mat res, bool &flag)  {  Mat temps(66,66,CV\_8UC1,Scalar(0));  temps=src.clone();  flag=false;  for(int j=1; j<=temps.cols-2; j++)  {  for(int i=1; i<=temps.rows-2; i++)  {  if(P.at<uchar>(i,j)!=p)  temps.at<uchar>(i,j)=temps.at<uchar>(i,j);  else  {  int x[9];  x[0]=temps.at<uchar>(i,j);  x[1]=temps.at<uchar>(i,j+1);  x[2]=temps.at<uchar>(i-1,j);  x[3]=temps.at<uchar>(i,j-1);  x[4]=temps.at<uchar>(i+1,j);  x[5]=temps.at<uchar>(i+1,j+1);  x[6]=temps.at<uchar>(i-1,j+1);  x[7]=temps.at<uchar>(i-1,j-1);  x[8]=temps.at<uchar>(i+1,j-1);  int a1=hs(x[0],x[1],x[6],x[2]);  int a2=hs(x[0],x[2],x[7],x[3]);  int a3=hs(x[0],x[3],x[8],x[4]);  int a4=hs(x[0],x[4],x[5],x[1]);  int t=fs(a1,a2,a3,a4,x[0]);  if(t==0)  flag=true;  temps.at<uchar>(i,j)=t;  }  }  }  temps.copyTo(res);  } |

1. Thinning Operator

按三個步驟：

（1）Mark-Interior/Border-Pixel

（2）The pair relationship

（3）Connected shrink

循環直到shrink后的結果不會變化。

|  |
| --- |
| bool flag=true;  Mat temp(66,66,CV\_8UC1,Scalar(0));  imgDownSample.copyTo(temp);  while(flag)  {  //yokoi  Mat YokoiNumber(66,66,CV\_8UC1,Scalar(6));  Yokoi(temp, YokoiNumber);  //pair relationship  Mat PairRelationship(66,66,CV\_8UC1,Scalar(0));  Pair(YokoiNumber, PairRelationship);  //shrink  Mat ConnectedShrink(66,66,CV\_8UC1,Scalar(0));  Shrink(temp, PairRelationship, ConnectedShrink, flag);  ConnectedShrink.copyTo(temp);  }    Mat imgThinning(66,66,CV\_8UC1,Scalar(0));  temp.copyTo(imgThinning); |

Thinning的結果：

