Computer and Robot Vision

Homework#7

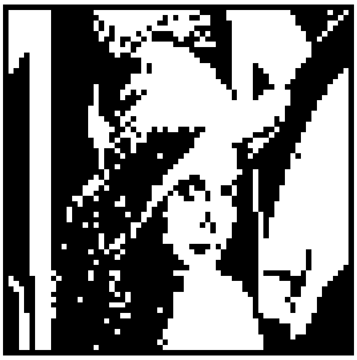
R01944040 柳成蔭

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。

(a) Yokoi Connectivity Number

Formula h

- 4-connectivity

$$h(b,c,d,e) = \begin{cases} q & \text{if } b = c \text{ and } (d \neq b \text{ or } e \neq b) \\ r & \text{if } b = c \text{ and } (d = b \text{ and } e = b) \\ s & \text{if } b \neq c \end{cases}$$

- q: corner $1 \rightarrow 0$ transition
- r: corner all 1 , no transition
- s: center 1 , neighbor 0 , nothing will happen

```
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

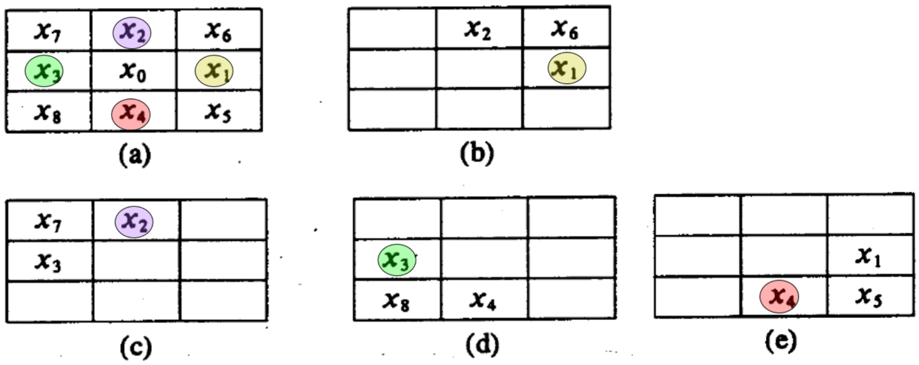
$$f(a_1,a_2,a_3,a_4) = \begin{cases} 5 & \text{if } a_1 = a_2 = a_3 = a_4 = r \\ n & \text{where } n = \#\{a_k | a_k = q\}, \text{ otherwise} \end{cases}$$

- 5: no transition all 8 neighbors 1, thus interior
- n: 1 transition generates one connected component if center removed

```
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

corner neighborhood



4- Connectivity number

$$y = f(a_1, a_2, a_3, a_4)$$

$$a_1 = h(x_0, x_1, x_6, x_2)$$

$$a_2 = h(x_0, x_2, x_7, x_3)$$

$$a_3 = h(x_0, x_3, x_8, x_4)$$

$$a_4 = h(x_0, x_4, x_5, x_1)$$

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;
            }
        }
    }
}
```

(b) Pair Relationship

Formula hp

h: determines whether first argument equals label 1

$$h(a,1) = \begin{cases} 1 & \text{if } a = 1 \\ 0 & \text{otherwise} \end{cases}$$

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

Formula yp

4-connected mode, output y

$$y = \begin{cases} q & \text{if } \sum_{n=1}^{4} h(x_n,1) < 1 \text{ or } x_0 \neq 1 \\ p & \text{if } \sum_{n=1}^{4} h(x_n,1) \geq 1 \text{ and } x_0 = 1 \end{cases}$$

q : not deletable if Yokoi number $\neq 1$ or no neighbor 1

p : possibly deletable if Yokoi number 1 and some neighbor 1

```
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);
        }
    }
}
```

(c) Connected Shrink Operator

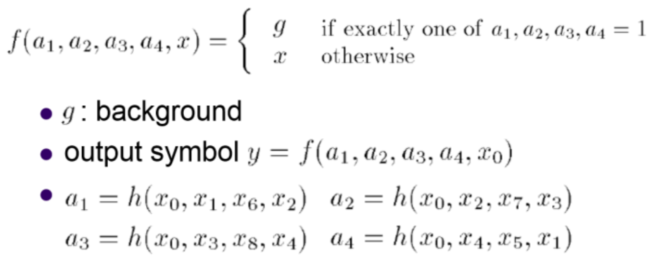
Formula hs

- h: determines whether corner connected

$$h(b,c,d,e) = \begin{cases} 1 & \text{if } b = c \text{ and } (d \neq b \text{ or } e \neq b) \\ 0 & \text{otherwise} \end{cases}$$

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

Formula fs



$$f(a_1, a_2, a_3, a_4, x) = \begin{cases} g & \text{if exactly one of } a_1, a_2, a_3, a_4 = 1 \\ x & \text{otherwise} \end{cases}$$

- g : background
- output symbol $y = f(a_1, a_2, a_3, a_4, x_0)$
- $a_1 = h(x_0, x_1, x_6, x_2)$ $a_2 = h(x_0, x_2, x_7, x_3)$
  $a_3 = h(x_0, x_3, x_8, x_4)$ $a_4 = h(x_0, x_4, x_5, x_1)$

```
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);
}
```

(d) 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: