Algorithm & Principal code fragment:

#include <iostream>

#include <opencv2/core/core.hpp>

#include <opencv2/highgui/highgui.hpp>

#include <cv.h>

using namespace std;

using namespace cv;

Mat picInitialSrc(Mat pic, Mat src, int row, int col)

{

int i,j;

for(i=0; i<col; i++)

{

for(j=0; j<row; j++)

{

pic.at<unsigned char>(i,j) = src.at<unsigned char>(i,j);

}

}

return pic;

}

Mat dilation(Mat dila, Mat src, int row, int col)

{

int i,j;

for(i=0; i<col; i++)

{

for(j=0; j<row; j++)

{

if(i>=2 && j>=1 ) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i-2,j-1)) dila.at<unsigned char>(i-2,j-1) = src.at<unsigned char>(i,j); }

if(i>=2) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i-2,j )) dila.at<unsigned char>(i-2,j) = src.at<unsigned char>(i,j); }

if(i>=2 && j<=511) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i-2,j+1)) dila.at<unsigned char>(i-2,j+1) = src.at<unsigned char>(i,j); }

if(i>=1 && j>=2) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i-1,j-2)) dila.at<unsigned char>(i-1,j-2) = src.at<unsigned char>(i,j); }

if(i>=1 && j>=1) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i-1,j-1)) dila.at<unsigned char>(i-1,j-1) = src.at<unsigned char>(i,j); }

if(i>=1) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i-1,j )) dila.at<unsigned char>(i-1,j) = src.at<unsigned char>(i,j); }

if(i>=1 && j<=511) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i-1,j+1)) dila.at<unsigned char>(i-1,j+1) = src.at<unsigned char>(i,j); }

if(i>=1 && j<=510) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i-1,j+2)) dila.at<unsigned char>(i-1,j+2) = src.at<unsigned char>(i,j); }

if(j>=2) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i ,j-2)) dila.at<unsigned char>(i,j-2) = src.at<unsigned char>(i,j); }

if(j>=1) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i ,j-1)) dila.at<unsigned char>(i,j-1) = src.at<unsigned char>(i,j); }

{ if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i ,j )) dila.at<unsigned char>(i,j) = src.at<unsigned char>(i,j); }

if(j<=511) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i ,j+1)) dila.at<unsigned char>(i,j+1) = src.at<unsigned char>(i,j); }

if(j<=510) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i ,j+2)) dila.at<unsigned char>(i,j+2) = src.at<unsigned char>(i,j); }

if(i<=511 && j>=2) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i+1,j-2)) dila.at<unsigned char>(i+1,j-2) = src.at<unsigned char>(i,j); }

if(i<=511 && j>=1) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i+1,j-1)) dila.at<unsigned char>(i+1,j-1) = src.at<unsigned char>(i,j); }

if(i<=511) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i+1,j )) dila.at<unsigned char>(i+1,j) = src.at<unsigned char>(i,j); }

if(i<=511 && j<=511){ if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i+1,j+1)) dila.at<unsigned char>(i+1,j+1) = src.at<unsigned char>(i,j); }

if(i<=511 && j<=510){ if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i+1,j+2)) dila.at<unsigned char>(i+1,j+2) = src.at<unsigned char>(i,j); }

if(i<=510 && j>=1) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i+2,j-1)) dila.at<unsigned char>(i+2,j-1) = src.at<unsigned char>(i,j); }

if(i<=510 ) { if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i+2,j )) dila.at<unsigned char>(i+2,j) = src.at<unsigned char>(i,j); }

if(i<=510 && j<=511){ if(src.at<unsigned char>(i,j) > dila.at<unsigned char>(i+2,j+1)) dila.at<unsigned char>(i+2,j+1) = src.at<unsigned char>(i,j); }

}

}

return dila;

}

Mat erosion(Mat eros, Mat src, int row, int col)

{

int i,j;

//Mat keyPoint;

//keyPoint = imread("lena.bmp",CV\_LOAD\_IMAGE\_GRAYSCALE);

int keyPoint[512][512] = {0};

//Mat keyPoint(row, col, CV\_8U);

//keyPoint = picInitialSrc(keyPoint, src, row, col);

for(i=0; i<col; i++)

{

for(j=0; j<row; j++)

{

if( ( src.at<unsigned char>(i-2,j-1) != 0 && i>=2 && j>=1 ) &&

( src.at<unsigned char>(i-2,j) != 0 && i>=2 ) &&

( src.at<unsigned char>(i-2,j+1) != 0 && i>=2 && j<=511 ) &&

( src.at<unsigned char>(i-1,j-2) != 0 && i>=1 && j>=2 ) &&

( src.at<unsigned char>(i-1,j-1) != 0 && i>=1 && j>=1 ) &&

( src.at<unsigned char>(i-1,j) != 0 && i>=1 ) &&

( src.at<unsigned char>(i-1,j+1) != 0 && i>=1 && j<=511 ) &&

( src.at<unsigned char>(i-1,j+2) != 0 && i>=1 && j<=510 ) &&

( src.at<unsigned char>(i,j-2) != 0 && j>=2 ) &&

( src.at<unsigned char>(i,j-1) != 0 && j>=1 ) &&

( src.at<unsigned char>(i,j) != 0 ) &&

( src.at<unsigned char>(i,j+1) != 0 && j<=511 ) &&

( src.at<unsigned char>(i,j+2) != 0 && j<=510 ) &&

( src.at<unsigned char>(i+1,j-2) != 0 && i<=511 && j>=2 ) &&

( src.at<unsigned char>(i+1,j-1) != 0 && i<=511 && j>=1 ) &&

( src.at<unsigned char>(i+1,j) != 0 && i<=511 ) &&

( src.at<unsigned char>(i+1,j+1) != 0 && i<=511 && j<=511 ) &&

( src.at<unsigned char>(i+1,j+2) != 0 && i<=511 && j<=510 ) &&

( src.at<unsigned char>(i+2,j-1) != 0 && i<=510 && j>=1 ) &&

( src.at<unsigned char>(i+2,j) != 0 && i<=510 ) &&

( src.at<unsigned char>(i+2,j+1) != 0 && i<=510 && j<=511 ) ){

keyPoint[i][j] = 255;// is keyPoint

}

else

{

keyPoint[i][j] = 0;//not keyPoint

}

}

}//end for

for(i=0; i<col; i++)

{

for(j=0; j<row; j++)

{

if(keyPoint[i][j] == 255)

{

if(src.at<unsigned char>(i-2,j-1) < src.at<unsigned char>(i-2,j))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i-2,j-1);

else

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i-2,j);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i-2,j+1))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i-2,j+1);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i-1,j-2))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i-1,j-2);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i-1,j-1))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i-1,j-1);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i-1,j))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i-1,j);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i-1,j+1))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i-1,j+1);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i-1,j+2))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i-1,j+2);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i,j-2))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i,j-2);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i,j-1))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i,j-1);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i,j))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i,j);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i,j+1))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i,j+1);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i,j+2))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i,j+2);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i+1,j-2))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i+1,j-2);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i+1,j-1))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i+1,j-1);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i+1,j))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i+1,j);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i+1,j+1))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i+1,j+1);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i+1,j+2))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i+1,j+2);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i+2,j-1))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i+2,j-1);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i+2,j))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i+2,j);

if(eros.at<unsigned char>(i,j) > src.at<unsigned char>(i+2,j+1))

eros.at<unsigned char>(i,j) = src.at<unsigned char>(i+2,j+1);

}

else

{

eros.at<unsigned char>(i,j) = 0;

}

}

}

return eros;

}

Mat erosionA(Mat eros, Mat src, int row, int col)

{

int i,j;

for(i=0; i<col; i++)

{

for(j=0; j<row; j++)

{

if( ( src.at<unsigned char>(i,j-1) == 0 && j >= 1 ) ||

( src.at<unsigned char>(i+1,j) == 0 && i <= 511 ) ||

( src.at<unsigned char>(i,j) == 0 ) ){

eros.at<unsigned char>(i,j) = 0;

}

else

{

eros.at<unsigned char>(i,j) = 255;

}

}

}

return eros;

}

Mat erosionB(Mat eros, Mat src, int row, int col)

{

int i,j;

for(i=0; i<col; i++)

{

for(j=0; j<row; j++)

{

if( ( src.at<unsigned char>(i-1,j) == 0 && i >= 1 ) ||

( src.at<unsigned char>(i-1,j+1) == 0 && i >= 1 && j <= 511 ) ||

( src.at<unsigned char>(i,j+1) == 0 && j <= 511 ) ){

eros.at<unsigned char>(i,j) = 0;

}

else

{

eros.at<unsigned char>(i,j) = 255;

}

}

}

return eros;

}

Mat opening(Mat src, int row, int col)

{

Mat temp(row, col, CV\_8U);

Mat temp2(row, col, CV\_8U);

Mat temp3(row, col, CV\_8U);

temp = picInitialSrc(temp, src, row, col);

temp2 = picInitialSrc(temp2, src, row, col);

temp3 = picInitialSrc(temp3, src, row, col);

temp = dilation(temp, src, row, col);

temp2 = picInitialSrc(temp2, temp, row, col);

temp3 = erosion(temp3, temp2, row, col);

return temp3;

}

Mat closing(Mat src, int row, int col)

{

Mat temp(row, col, CV\_8U);

Mat temp2(row, col, CV\_8U);

Mat temp3(row, col, CV\_8U);

temp = picInitialSrc(temp, src, row, col);

temp2 = picInitialSrc(temp2, src, row, col);

temp3 = picInitialSrc(temp3, src, row, col);

temp = erosion(temp, src, row, col);

temp2 = picInitialSrc(temp2, temp, row, col);

temp3 = dilation(temp2, temp, row, col);

return temp3;

}

void showSavePicture(Mat picture,string windowName,string saveName)

{

namedWindow( windowName, WINDOW\_AUTOSIZE );

imshow( windowName, picture );

imwrite( saveName , picture );

}

Mat hit\_and\_miss(Mat src,int row,int col)

{

Mat src\_inverse(row, col, CV\_8U);

Mat A(row, col, CV\_8U);

Mat B(row, col, CV\_8U);

Mat HAM(row, col, CV\_8U);

int i,j;

for(int i=0; i<col; i++)

{

for(int j=0; j<row; j++)

{

if(src.at<unsigned char>(i,j) == 0)

src\_inverse.at<unsigned char>(i,j) = 255;

else

src\_inverse.at<unsigned char>(i,j) = 0;

}

}

A = erosionA(A,src,row,col);

B = erosionB(B,src\_inverse,row,col);

showSavePicture(A,"A","A.bmp");

showSavePicture(B,"B","B.bmp");

for(int i=0; i<col; i++)

{

for(int j=0; j<row; j++)

{

if(A.at<unsigned char>(i,j) == 255 && B.at<unsigned char>(i,j) == 255)

HAM.at<unsigned char>(i,j) = 255;

else

HAM.at<unsigned char>(i,j) = 0;

}

}

return HAM;

}

int main(int argc,char\*\* argv )

{

Mat src;

src = imread("lena.bmp",CV\_LOAD\_IMAGE\_GRAYSCALE);

int i,j;

int row = src.rows;

int col = src.cols;

Mat dila(row, col, CV\_8U);

Mat eros(row, col, CV\_8U);

Mat opic(row, col, CV\_8U);

Mat clos(row, col, CV\_8U);

//Mat HAM (row, col, CV\_8U);

dila = picInitialSrc(dila,src,row,col);

eros = picInitialSrc(eros,src,row,col);

opic = picInitialSrc(opic,src,row,col);

clos = picInitialSrc(clos,src,row,col);

dila = dilation(dila, src, row, col);

eros = erosion(eros, src, row, col);

opic = opening(src, row, col);

clos = closing(src, row, col);

//HAM = hit\_and\_miss(src, row, col);

showSavePicture(src,"src","src.bmp");

showSavePicture(dila,"dilation","dilation.bmp");

showSavePicture(eros,"erosion","erosion.bmp");

showSavePicture(opic,"closing","closing.bmp");

showSavePicture(clos,"openning","opening.bmp");

//showSavePicture(HAM,"hit-and-miss","hit\_and\_miss.bmp");

//opic = dilation(eros, eros,row,col);

//showSavePicture(eros,"opening","opening.bmp");

//clos = erosion(dila, dila,row, col);

//showSavePicture(clos,"closing","closing.bmp");

waitKey(0);

return 0;

}

Parameters:

Mat src;

src = imread("lena.bmp",CV\_LOAD\_IMAGE\_GRAYSCALE);

int i,j;

int row = src.rows;

int col = src.cols;

Mat dila(row, col, CV\_8U);

Mat eros(row, col, CV\_8U);

Mat opic(row, col, CV\_8U);

Mat clos(row, col, CV\_8U);

Resulting Images:



dilation erosion



opening closing