P78-2

缩放函数文件myScale.m：

function g = myScale(f, varargin)

k1 = varargin{1};

if nargin==2

k2 = k1;

else

k2 = varargin{2};

end

[m,n] = size(f);

M = round(m\*k1);

N = round(n\*k2);

g = zeros(M, N, 'uint8');

for i=1:M-1

for j=1:N-1

g(i, j) = f(round(double(i)/k1), round(double(j)/k2));

end

end

g(M, N) = f(floor(double(i)/k1), floor(double(j)/k2));

测试文件scaleDemo.m:

k = 0.5;

f = imread('img/Fig0219(a).tif');

[m, n] = size(f);

% 裁剪图片

f = f(1:round(k\*m), 1:round(k\*n));

k1 = 1.3;

g1 = myScale(f, k1);

k2 = 0.7;

g2 = myScale(f, k2);

k31 = 0.7;

k32 = 1.3;

g3 = myScale(f, k31, k32);

figure();

imshow(f);

title('原图');

figure();

imshow(g1, []);

title('放大');

figure();

imshow(g2);

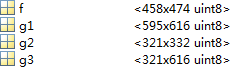
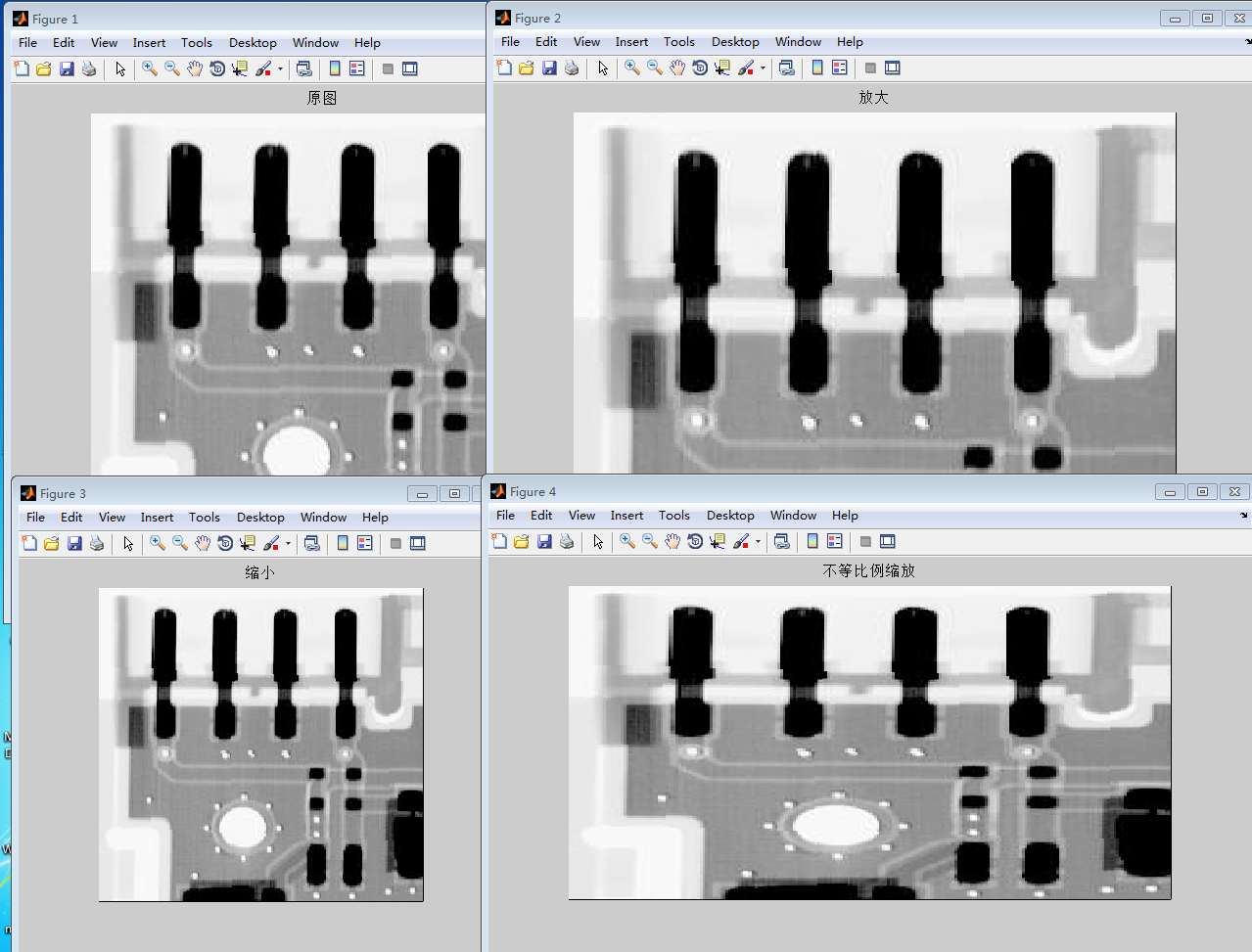
title('缩小');

figure();

imshow(g3);

title('不等比例缩放');

测试结果：



P97-1

P阈值分割函数文件pThreshold.m:

function g = pThreshold( f, idealP, errLimit )

[m,n] = size(f);

p = zeros(256,1,'uint16');

pRate = zeros(256, 1);

P = zeros(256,1);

for i=1:m

for j=1:n

p(f(i,j)+1) = p(f(i,j)+1)+1;

end

end

pRate(1) = double(p(1))/double(m\*n);

P(1) = pRate(1);

for i=2:256

pRate(i) = double(p(i))/double(m\*n);

P(i) = P(i-1)+pRate(i);

if abs(P(i)-idealP)<=errLimit

break;

end

end

tmp = i;

g = zeros(m, n, 'uint8');

for i=1:m

for j=1:n

if f(i,j)<tmp

g(i,j) = 0;

else

g(i,j) = 255;

end

end

end

end

峰谷阈值分割文件pkvlThreshold.m:

function g = pkvlThreshold( f )

%峰谷阈值分割图像

[m,n] = size(f);

%灰度值个数

pOrig = zeros(256,1);

for i=1:m

for j=1:n

pOrig(f(i,j)+1) = pOrig(f(i,j)+1)+1.0;

end

end

figure();

subplot(2,2,1);

area(pOrig,'DisplayName','pOrig');

title('原直方图');

%采用中值过滤处理灰度直方图，砍掉小的峰和谷

p = pOrig;

range=uint16(20);

for i=1:range

tmp = p(1:i+range);

[m1, n1] = size(tmp);

tmp = bubbleSort(m1, tmp);

p(i) = tmp(uint8(m1+1)/2);

end

for i=(range+1):(256-range)

tmp = p(i-range:i+range);

[m1, n1] = size(tmp);

tmp = bubbleSort(m1, tmp);

p(i) = tmp(uint8(m1+1)/2);

end

for i=(256-range):256

tmp = p(i-range:256);

[m1, n1] = size(tmp);

tmp = bubbleSort(m1, tmp);

p(i) = tmp(uint8(m1+1)/2);

end

subplot(2,2,2);

area(p ,'DisplayName','p');

title('中值过滤后的直方图');

toBeAValley=0;

valleyCount=0;

valleys = zeros(256,1,'uint8');

%计算对应的增减趋势，以及谷底

grad = zeros(256,1,'int8');

for i=2:256

diff = p(i)-p(i-1);

if diff>0

grad(i) = 1;

%谷底选择的是谷底最右侧的点

if toBeAValley==1

valleyCount=valleyCount+1;

valleys(valleyCount)=i;

toBeAValley=0;

end

elseif diff<0

grad(i) = -1;

toBeAValley=1;

else

grad(i) = 0;

end

end

grad(1)=grad(2);

subplot(2,2,3);

area(grad,'DisplayName','grad');

title('增长趋势图');

%每个谷底之前的灰度值的映射表（考虑到用于多于两个峰的情况）

greys = zeros(valleyCount+1, 1, 'uint8');

grepPerValley = 255.0/double(valleyCount);

for i=1:valleyCount+1

greys(i) = (i-1)\*grepPerValley;

end

g = zeros(m, n, 'uint8');

hasSet=0;

for i=1:m

for j=1:n

for k=1:valleyCount

if f(i,j)+1<valleys(k)

g(i,j) = greys(k);

hasSet=1;

break;

end

end

if hasSet==0

g(i,j) = greys(valleyCount+1);

else

hasSet=0;

end

end

end

end

中值过滤处理中的bubbleSort函数定义在冒泡排序函数文件bubbleSort.m中:

function [f, g] = bubbleSort( n, varargin )

% 参数1为要排序的个数

% 参数2~3个

% 参数2为要排序的数组

% 参数3伴随参数2排序

error(nargchk(2,3,nargin));

f = varargin{1};

if nargin==2

for i=1:n-1

for j=2:n-i+1

if f(j-1)>f(j)

t=f(j);

f(j)=f(j-1);

f(j-1)=t;

end

end

end

else

g= varargin{2};

for i=1:n-1

for j=2:n-i+1

if f(j-1)>f(j)

t=f(j);

f(j)=f(j-1);

f(j-1)=t;

t=g(j);

g(j)=g(j-1);

g(j-1)=t;

end

end

end

end

end

测试文件separate.m：

rows = 2;

cols = 2;

f = imread('img/Fig0309(a).tif');

p = 0.45;

errLimit = 0.01;

g1 = pThreshold(f, p, errLimit);

g2 = pkvlThreshold(f);

figure();

subplot(rows, cols, 1);

imshow(f);

title('原图');

subplot(rows, cols, 3);

imshow(g1);

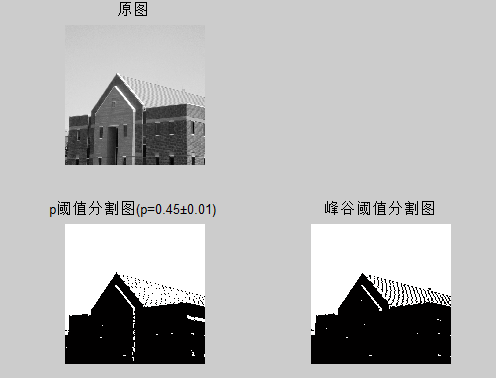
title('p阈值分割图(p=0.25±0.01)');

subplot(rows, cols, 4);

imshow(g2);

title('峰谷阈值分割图');

测试结果：



峰谷阈值分割函数中产生的的直方图处理图：

