% 输入一幅图像，找出

clc;

clear;

% -----------------------------------------------------------------------------------------------------------

% 坐标轴：

% 1。imshow图像是按矩阵显示（第一个下标为行，第二个下标为列）

% 2。plot本来是以左下角为坐标原点，右向为x轴正向，上方为y轴正向的。

% 3。如果在imshow出来的图形上用plot，则坐标轴的右方为x轴正向，下方为y轴正向，刚好分别对应和图像的列和行。

% 读图像与设置初始参数

t=rgb2gray(imread('SnakeTest.bmp'));

I = im2double(t);%fly\_gray.jpg

imshow(I);

hold on;

axis on;

% 图像的大小

sizeX = size(I,2);

sizeY = size(I,1);

% 图像的梯度

[gIx,gIy]=gradient(I);

I\_grad = zeros(sizeX,sizeY); % 梯度的模，先一次性计算出来，为了计算能量所用，而且调整了坐标，使得和plot函数的坐标一致

for xi=1:sizeX

for yi=1:sizeY

I\_grad(xi,yi) = sqrt(gIx(yi,xi)^2 + gIy(yi,xi)^2);

end

end

VCount = 200; % 轮廓上点的个数

V\_init = zeros(VCount+2,2); % V是要找的轮廓，共有VCount个点，（V(i,1),V(i,2)）是轮廓上第i个点的坐标

% 初始化轮廓为一个圆（顺时针的顺序）

R =30;%150

for vi=1:VCount

V\_init(vi+1,1) = floor(130 + R \* cos(vi \* 2\*3.14/VCount));

V\_init(vi+1,2) = floor(140 + R \* sin(vi \* 2\*3.14/VCount));

end

% 超出边界的部分赋值为边界

V\_init(find(V\_init(:,1)>sizeX),1) = sizeX-4;

V\_init(find(V\_init(:,1)<1),1) = 2;

V\_init(find(V\_init(:,2)>sizeY),2) = sizeY-4;

V\_init(find(V\_init(:,2)<1),2) = 2;

V\_init(1,:) = V\_init(VCount+1,:);

V\_init(VCount+2,:) = V\_init(2,:);

hold on;

plot(V\_init(:,1),V\_init(:,2),'b');

plot(V\_init(:,1),V\_init(:,2),'b.');

plot(V\_init(2,1),V\_init(2,2),'b\*');

plot(V\_init(3,1),V\_init(3,2),'b\*');

% 根据Snake算法寻找轮廓

repeatK = 50; % 迭代次数

minE = 10000; % 初始化的最小能量

V\_min = V\_init; % 当前最小能量对应的轮廓

[I\_gradSort,I\_gradIndex] = sort(I\_grad(:));

avgGrad = mean(I\_gradSort((length(I\_gradSort)-floor(length(I\_gradSort)/50)):length(I\_gradSort)));

% 轮廓上大概的平均梯度，梯度最大的1/50的点的平均

for k = 1:repeatK

vi=2; avgCon=0;

while vi <= VCount

viDist = norm(V\_min(vi+1,:)-V\_min(vi,:));

% 如果两个点距离的很近，则合并

if viDist <= 3

for vii = (vi+1):(VCount+1)

V\_min(vii,:) = V\_min(vii+1,:);

end

V\_min(VCount+2,:) = [];

VCount = VCount - 1;

elseif viDist >= 7 % 如果两个点距离的很远，则中间插入一点

for vii = (VCount+1):-1:(vi+1)

V\_min(vii+1,:) = V\_min(vii,:);

end

V\_min(vi+1,:) = floor((V\_min(vi,:) + V\_min(vi+2,:))/2);

VCount = VCount + 1;

V\_min(1,:) = V\_min(VCount+1,:);

V\_min(VCount+2,:) = V\_min(2,:);

else

avgCon = avgCon + viDist;

end

vi = vi + 1;

end

avgCon = avgCon / VCount; % 所有点之间的平均距离，规范化所用

% 每次循环把所有点都向周围8个点移动一次，根据局部能量调整轮廓

for vi = 2:(VCount+1)

vi\_01 = (V\_min(vi,:) - V\_min(vi-1,:)) / norm(V\_min(vi,:) - V\_min(vi-1,:));

vi\_12 = (V\_min(vi+1,:) - V\_min(vi,:)) / norm(V\_min(vi+1,:) - V\_min(vi,:));

theta = acos( sum(vi\_01 .\* vi\_12) );

% 如果两个向量的夹角超过90度，则直接取中间的点

if theta > 3.14/2

V\_min(vi,:) = floor((V\_min(vi-1,:) + V\_min(vi+1,:))/2);

% 防止重复的点

if V\_min(vi,1)==V\_min(vi-1,1) && V\_min(vi,2)==V\_min(vi-1,2)

V\_min(vi,1) = V\_min(vi,1) - 1;

end

if V\_min(vi,1)==V\_min(vi+1,1) && V\_min(vi,2)==V\_min(vi+1,2)

V\_min(vi,2) = V\_min(vi,2) - 1;

end

V\_min(1,:) = V\_min(VCount+1,:);

V\_min(VCount+2,:) = V\_min(2,:);

continue;

end

% 该点的切线方向（顺时针方向，右边是轮廓内部）

t\_vi = (V\_min(vi,:) - V\_min(vi-1,:)) / norm(V\_min(vi,:) - V\_min(vi-1,:)) + (V\_min(vi+1,:) - V\_min(vi,:)) / norm(V\_min(vi+1,:) - V\_min(vi,:));

% 法线方向（把切线逆时针旋转90度）

n\_vi = [t\_vi(2),-t\_vi(1)];

n\_vi = n\_vi / norm(n\_vi);

Nei\_vi = snake\_Neighbor(sizeX,sizeY,V\_min(vi,1),V\_min(vi,2),V\_min); % 得到轮廓上某点周围的点的坐标（8个）

Nei\_Egrad = zeros(1,size(Nei\_vi,1)); % 第vi个点的领域的所有点的梯度（外部能量）

Nei\_Econ = zeros(1,size(Nei\_vi,1)); % 平滑性约束（内部能量）

Nei\_Ebal = zeros(1,size(Nei\_vi,1)); % 气球约束（内部能量）

for ni=1:size(Nei\_vi,1)

Nei\_Egrad(ni) = -I\_grad(Nei\_vi(ni,1),Nei\_vi(ni,2)); % 外部能量（梯度）

Nei\_Econ(ni) = norm(Nei\_vi(ni,:) - (V\_min(vi-1,:) + V\_min(vi+1,:))/2); % 平滑性约束

% 邻域的点和原来的点在法线方向上的增长度量（前面的系数为正，要使得能量尽量小，则有往轮廓外扩张的趋势）

Nei\_Ebal(ni) = - sum( n\_vi .\* (V\_min(vi,:)-Nei\_vi(ni,:)) );

end

Nei\_Egrad\_norm = Nei\_Egrad/avgGrad;

Nei\_Econ\_norm = Nei\_Econ/avgCon;

if (max(Nei\_Ebal)-min(Nei\_Ebal))~=0

vin\_grad = (I\_grad(V\_min(vi,1),V\_min(vi,2)) + I\_grad(V\_min(vi-1,1),V\_min(vi-1,2)) + I\_grad(V\_min(vi+1,1),V\_min(vi+1,2))) / 3;

Nei\_Ebal\_norm = (1 - vin\_grad/avgGrad) \* (Nei\_Ebal-min(Nei\_Ebal)) / (max(Nei\_Ebal)-min(Nei\_Ebal));

else

Nei\_Ebal\_norm = 0;

end

Nei\_E = Nei\_Econ\_norm + Nei\_Ebal\_norm +15\*Nei\_Egrad\_norm;

% 取具有最小能量的点

[Nei\_ESort,Nei\_EIndex] = sort(Nei\_E);

V\_min(vi,:) = Nei\_vi(Nei\_EIndex(1),:);

V\_min(1,:) = V\_min(VCount+1,:);

V\_min(VCount+2,:) = V\_min(2,:);

end

if mod(k,10)==0

plot(V\_min(:,1),V\_min(:,2),'r');

plot(V\_min(:,1),V\_min(:,2),'r.');

end

end

% -----------------------------------------------------------------------------------------------------------

% 画出结果轮廓

plot(V\_min(:,1),V\_min(:,2),'y');

plot(V\_min(:,1),V\_min(:,2),'y.');

% ------------------------------------------------------------------------------------------------------------