Report on the silent object detection

Pengyu Wang

Step1 Input image.

Program:

```
clc;

clear;

RGB=imread('4.jpg');
```

Explanation

I choose this image

Step1-1 Extract image information

Program

```
%将 RGB 图像分为三个通道的灰度图

R=RGB(:,:,1);

G=RGB(:,:,2);

B=RGB(:,:,3);

%将灰度范围压缩到 0-32

R=R/8;

G=G/8;

B=B/8;

%将像素数值类型从 uint8 转化为 double

R=double(R);

G=double(G);

B=double(B);

%将三通道颜色通过单数值表示,并显示单值表示后的图像

RGB2=R*32*32+G*32+B;          %RGB2 为颜色单值表示后的一维矩阵

figure,imshow(imdivide(RGB2,max(max(RGB2))));
```

Explanation

Firstly, I use three matrices(R,G,B) to store the three channels of the rgb image. Secondly, I divide each matrix with 8 to reduce the number of color. Then I turn the data form from

uint8 to double to avoid overflow in the following step. Finally I regard the color vector (R,G,B) as a hexadecimal number, so I choose the formula RGB2=R*32*32+G*32+B to express all the colors in one matrix(RGB2) and show this matrix in gray image.

Output

Step1-2 Segment the input image to superpixels

Program

```
%step1-2

%使用 vlfeat 产生超像素分割矩阵

imlab=vl_xyz2lab(vl_rgb2xyz(RGB));

imlab=single(imlab);
```

```
segments=vl_slic(imlab,30,0.1);

num_sup=max(max(segments));

[m n] = size(segments);

[sx,sy]=vl_grad(double(segments), 'type', 'forward') ;

 s = find(sx | sy) ;

 imp = RGB ;

 imp([s s+numel(imp(:,:,1)) s+2*numel(imp(:,:,1))]) = 0 ;

o = imp;

figure,imshow(o);
```

Explanation

In this step, I choose the function vl_slic to segment the input image to superpixels. Firstly, I turn the rgb image to the lab form with function vl_rgb2xyz and vl_xyz2lab. Then I change the data form of lab image to single to fit the request of the function vl_slic. Finally I segment the input image to superpixels with the vl_slic and show this matrix(segments) in appropriate gray level range.

Output

Step2 Compute features of each superpixel

Program

```
%step2

num_bin=512;

RGB3=zeros(num_sup,[],'double');          %RGB3 第 i+1 行为第 i 块超像素包含的像素( 颜色分布 )

his=zeros(num_sup,num_bin);

for i=0:num_sup

    j=1;

    for x=1:m

        for y=1:n
```

```
            if segments(x,y)==i
                RGB3(i+1,j)=RGB2(x,y);
                j=j+1;
            end
        end
    end
    his(i+1,[1:num_bin])=hist(RGB3(i+1,1:j-1),num_bin)/(j+1);
end
```

Explanation

Firstly, I use a three-level circular to store pixels of each superpixel in a matrix(RGB3). In the matrix, the pixels in each row are the pixels from each superpixel. Then I compute the color histogram of each superpixel through computing the histogram(stored in the matrix his) of each row in the matrix RGB3 with the function hist and normalizing it.

Step3 Compute superpixel feature contrast

Program

```
%step3
%计算每个超像素的全局对比度

feature=zeros(1,num_sup+1,'double');
for i=0:num_sup
```

```
    for x=1:256
        for y=1:num_sup+1
            a=his(i+1,x)-his(y,x);
            b=his(i+1,x);
            if b>eps
                feature(1,i+1)=feature(1,i+1)+a*a/b;
            end
        end
    end
End
```

Explanation

In this step, I use a three-level circulate to compute the feature contrast(stored in matrix feature) of each segment through computing the contrast of the histogram stored in the matrix his. And in the circulate I use a if sentence to prevent dividing zero.

Step 4 Convert superpixel saliency to pixel saliency

Program

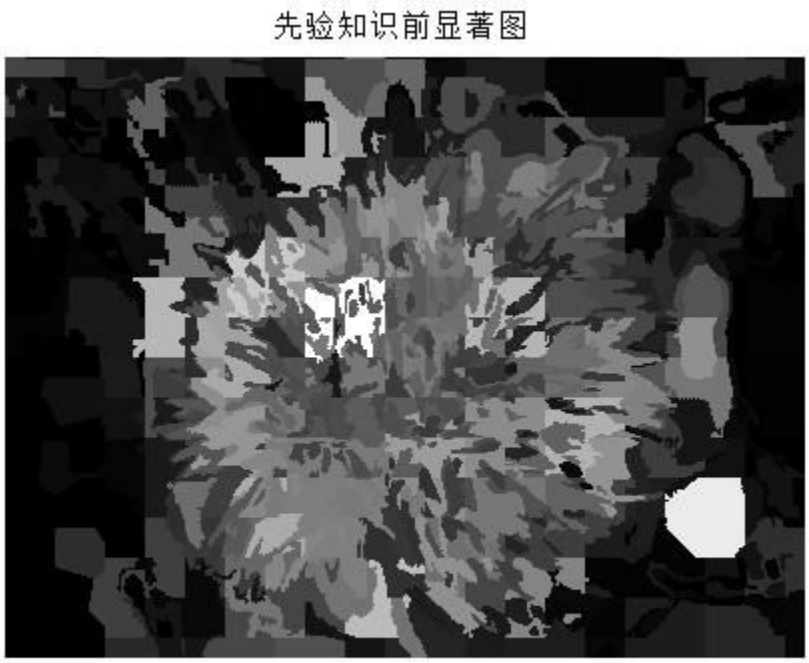
```
%step4
%将超像素全局对比度推广到像素对比度,fea_p 是推广后的图像,并以适当的灰度范围显示
fea_p=zeros(m,n,'double');
for x=1:m
```

```
    for y=1:n
        fea_p(x,y)=feature(1,segments(x,y)+1);
    end
end
figure,imshow(imdivide(fea_p,max(max(feature))));
```

Explanation

In this step I use a two-level circulate to put the segment contrast stored in matrix feature into a new image (fea_p) as the superpixel distribution and show this image in a appropriate gray level.

Output



Step5 Use priors to enhance the result

Program

```
%step5

%使用两种先验知识对提取结果进行优化

sigmaF = 6.2;

omega0 = 0.002;

sigmaD = 114;

sigmaC = 0.25;

coordinateMtx = zeros(m, n, 2);

coordinateMtx(:,:,1) = repmat((1:1:m)', 1, n);

coordinateMtx(:,:,2) = repmat(1:1:n, m, 1);

%center priors

centerY = m / 2;

centerX = n / 2;

centerMtx(:,:,1) = ones(m, n) * centerY;

centerMtx(:,:,2) = ones(m, n) * centerX;

SDMap = exp(-sum((coordinateMtx - centerMtx).^2,3) / sigmaD^2);

%color priors

Achannel=imlab(:,:,2);

Bchannel=imlab(:,:,3);
```

```
maxA = max(max(Achannel));

minA = min(min(Achannel));

normalizedA = (Achannel-minA )/ (maxA-minA);

maxB = max(max(Bchannel));

minB = min(min(Bchannel));

normalizedB = (Bchannel-minB) / (maxB - minB);

labDistSquare = normalizedA.^2 + normalizedB.^2;

SCMap = 1 - exp(-labDistSquare / (sigmaC^2));

VSMap = SDMap .* SCMap;


priormap=double(VSMap);

final=fea_p.*priormap;

figure,imshow((final-min(min(final)))/(max(max(final))-min(min(final)))),title('最后图像');
```

Explanation

In this step I use center priors and color priors to get the prior saliency map. And I multiply the saliency map got in the step4 and the prior map to get the final saliency map and show it in appropriate gray level.

Output

最后图像

Result

In this assignment, I use matlab to achieve the basic saliency detection. But there are some problems which haven’ t been resolved. The output is not so good, and there are some codes which can be simplify to reduce the runtime. What’ s more, because the lack of knowledge about C++, OpenCV isn’ t used in this assignment.