A REPORT ON THE SALIENT OBJECT DETECTION

Wenzong Wang

Explanation:

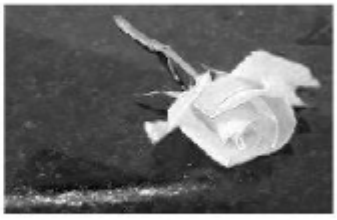
Step 1-1:

in this method superpixels are used to cut the image into several regions which are called as superpixels. To get the feature of the original image(color imformation), we divided the image into three images in R, G and B channels, then the matrix of the image was divided by 16 and we put the results together to let the maximum become 4095.

Here is the code for step 1-1:

```
clear all;
%Step 1-1:
I=imread('F:\matlab\flower.jpg');
Image_R=I(:,:,1);
Image_G=I(:,:,2);
Image_B=I(:,:,3);
figure,subplot(2,2,1);
imshow(I);
subplot(2,2,2);
imshow(Image_R);
subplot(2,2,3);
imshow(Image_G);
subplot(2,2,4);
imshow(Image_B);
```

Here is the result:



Step 1-2:

in this step the image is divided into many regions which are called as superpixels, theSLIC method is fast and effective, after doing this the pixels in the same region were labeled by the same number. In this step the function uint16 was used which is because the matrix of the original image is up to 255 and it is not enough for the 4095 kinds of colors.

Here is the code for step 1-2:

```
I3 = vl_xyz2lab(vl_rgb2xyz(I)) ;
I_single = single(I3);
segments = vl_slic(I_single,30, 0.1) ;
[sx,sy]=vl_grad(uint16(segments), 'type', 'forward') ;
  s = find(sx | sy) ;
  image_segments = I ;
  image_segments([s s+numel(I(:,:,1)) s+2*numel(I(:,:,1))]) = 0 ;
  %画出分割图
  figure;
  title('分割结果');
  imshow(image_segments);
```

Here is the result:

Step 2:

In the step1-1, the feature of color information was gotten and in the step1-2 the superpixels were also gotten. According to these we can get each histogram of every superpixel. In my program I used a circulation to finish this job.

Here is the code for step 2:

```
[m n]=size(segments);
feat=zeros(140,4096);
RGB4=zeros(140,[]);
hismatrix=zeros(140,10);
for x=0:139
    y=0;
    for i=1:m
        for j=1:n
            if segments(i,j)==x
                RGB4(x+1,y+1)=I2(i,j);
                y=y+1;
            end
        end
    end
    feat(x+1,[1,4096])=hist(RGB4(x+1,1:y-1),[1,4096]);
end
```

Step 3:

In this step we should compute the global regional contrast. The formulation has already given in the assignment, so what we should do is make the formulation into matlab code. In this part the circulation was used again.

Here is the code for step 3:

```
    num_sup=max(max(segments));
    feature=zeros(1,num_sup+1,'double');
    for i=0:num_sup
        for x=1:256
            for y=1:num_sup+1
                if feat(i+1,x)+feat(y,x)~=0

feature(1,i+1)=feature(1,i+1)+2*(feat(i+1,x)-feat(y,x))*(feat(i+1,x)-feat(y,x))/(feat(i+1,x)+feat(y,x));
                end
            end
```

```
        end
    end
```

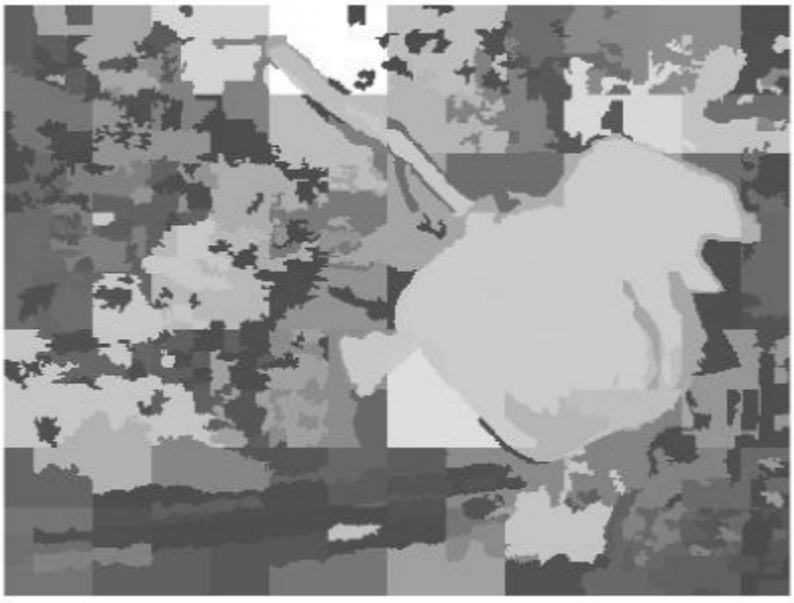
Step 4:

In this step we change the superpixel segmentation matrix into an initial saliency map, and the circulation was used in this step.

Here is the step for step 4:

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

Here is the result:



Step 5:

We used the method of SDSP which is a novel saliency detection. we choose the center prior for this step and finally we got a good result.

Here is the code for step 5:

```
sigmaF = 6.2;
omega0 = 0.002;
sigmaD = 114;
sigmaC = 0.25;
image=I;
%把图像转换到 Lab 色度域
[oriRows, oriCols, junk] = size(image);
image = double(image);
dsImage(:,:,1) = imresize(image(:,:,1), [256, 256],'bilinear');
dsImage(:,:,2) = imresize(image(:,:,2), [256, 256],'bilinear');
dsImage(:,:,3) = imresize(image(:,:,3), [256, 256],'bilinear');
lab = RGB2Lab(dsImage);

LChannel = lab(:,:,1);
AChannel = lab(:,:,2);
BChannel = lab(:,:,3);

LFFT = fft2(double(LChannel));
AFFT = fft2(double(AChannel));
BFFT = fft2(double(BChannel));

[rows, cols, junk] = size(dsImage);
LG = logGabor(rows,cols,omega0,sigmaF);
FinalLResult = real(ifft2(LFFT.*LG));
FinalAResult = real(ifft2(AFFT.*LG));
FinalBResult = real(ifft2(BFFT.*LG));

SFMap = sqrt(FinalLResult.^2 + FinalAResult.^2 + FinalBResult.^2);

%中心先验
coordinateMtx = zeros(rows, cols, 2);
coordinateMtx(:,:,1) = repmat((1:1:rows)', 1, cols);
coordinateMtx(:,:,2) = repmat(1:1:cols, rows, 1);

centerY = rows / 2;
centerX = cols / 2;
centerMtx(:,:,1) = ones(rows, cols) * centerY;
centerMtx(:,:,2) = ones(rows, cols) * centerX;
SDMap = exp(-sum((coordinateMtx - centerMtx).^2,3) / sigmaD^2);
maxA = max(AChannel(:));
minA = min(AChannel(:));
normalizedA = (AChannel - minA) / (maxA - minA);

maxB = max(BChannel(:));
```

```
minB = min(BChannel(:));
normalizedB = (BChannel - minB) / (maxB - minB);

labDistSquare = normalizedA.^2 + normalizedB.^2;
SCMap = 1 - exp(-labDistSquare / (sigmaC^2));
VSMap = SFMap .* SDMap .* SCMap;

VSMap =    imresize(VSMap, [oriRows, oriCols],'bilinear');
VSMap = uint8(mat2gray(VSMap) * 255);
prim=im2double(VSMap);
final=feat_p.*prim;
figure,imshow(imdivide(final,max(max(final))));
```

Here is the function of RGB2lab:

```
function labImage = RGB2Lab(image)

image = double(image);
normalizedR = image(:,:,1) / 255;
normalizedG = image(:,:,2) / 255;
normalizedB = image(:,:,3) / 255;

RSmallerOrEqualto4045 = normalizedR <= 0.04045;
RGreaterThan4045 = 1 - RSmallerOrEqualto4045;
tmpR = (normalizedR / 12.92) .* RSmallerOrEqualto4045;
tmpR = tmpR + power((normalizedR + 0.055)/1.055,2.4) .* RGreaterThan4045;

GSmallerOrEqualto4045 = normalizedG <= 0.04045;
GGreaterThan4045 = 1 - GSmallerOrEqualto4045;
tmpG = (normalizedG / 12.92) .* GSmallerOrEqualto4045;
tmpG = tmpG + power((normalizedG + 0.055)/1.055,2.4) .* GGreaterThan4045;

BSmallerOrEqualto4045 = normalizedB <= 0.04045;
BGreaterThan4045 = 1 - BSmallerOrEqualto4045;
tmpB = (normalizedB / 12.92) .* BSmallerOrEqualto4045;
tmpB = tmpB + power((normalizedB + 0.055)/1.055,2.4) .* BGreaterThan4045;

X = tmpR*0.4124564 + tmpG*0.3575761 + tmpB*0.1804375;
Y = tmpR*0.2126729 + tmpG*0.7151522 + tmpB*0.0721750;
Z = tmpR*0.0193339 + tmpG*0.1191920 + tmpB*0.9503041;

epsilon = 0.008856;   %actual CIE standard
kappa      = 903.3;           %actual CIE standard
```

```
Xr = 0.9642;%reference white D50
Yr = 1.0;           %reference white
Zr = 0.8251; %reference white

xr = X/Xr;
yr = Y/Yr;
zr = Z/Zr;

xrGreaterThanEpsilon = xr > epsilon;
xrSmallerOrEqualtoEpsilon = 1 - xrGreaterThanEpsilon;
fx = power(xr, 1.0/3.0) .* xrGreaterThanEpsilon;
fx = fx + (kappa*xr + 16.0)/116.0 .* xrSmallerOrEqualtoEpsilon;

yrGreaterThanEpsilon = yr > epsilon;
yrSmallerOrEqualtoEpsilon = 1 - yrGreaterThanEpsilon;
fy = power(yr, 1.0/3.0) .* yrGreaterThanEpsilon;
fy = fy + (kappa*yr + 16.0)/116.0 .* yrSmallerOrEqualtoEpsilon;

zrGreaterThanEpsilon = zr > epsilon;
zrSmallerOrEqualtoEpsilon = 1 - zrGreaterThanEpsilon;
fz = power(zr, 1.0/3.0) .* zrGreaterThanEpsilon;
fz = fz + (kappa*zr + 16.0)/116.0 .* zrSmallerOrEqualtoEpsilon;

[rows,cols,junk] = size(image);
labImage = zeros(rows,cols,3);
labImage(:,:,1) = 116.0 * fy - 16.0;
labImage(:,:,2) = 500.0 * (fx - fy);
labImage(:,:,3) = 200.0 * (fy - fz);
return;
```

Here is the function of logGabor:

```
function LG = logGabor(rows,cols,omega0,sigmaF)
        [u1, u2] = meshgrid(([1:cols]-(fix(cols/2)+1))/(cols-mod(cols,2)), ...
                                ([1:rows]-(fix(rows/2)+1))/(rows-mod(rows,2)));
        mask = ones(rows, cols);
        for rowIndex = 1:rows
              for colIndex = 1:cols
                    if u1(rowIndex, colIndex)^2 + u2(rowIndex, colIndex)^2 > 0.25
                          mask(rowIndex, colIndex) = 0;
                    end
              end
        end
        u1 = u1 .* mask;
```

```
        u2 = u2 .* mask;

        u1 = ifftshift(u1);
        u2 = ifftshift(u2);

        radius = sqrt(u1.^2 + u2.^2);
        radius(1,1) = 1;

        LG = exp((-(log(radius/omega0)).^2) / (2 * (sigmaF^2)));
        LG(1,1) = 0;
return;
```

Here is the result: