EE610: Assignment 3: Image Segmentation

Swapnil Bembde 14D070034

November 4, 2017

Feature Selection:

1. DCT Coefficients - Features obtained at the different points may not be the same but they can be the same in multi-dimensional space. Features based on DCT coefficients provides accurate and robust segmentation. For any image we don't have to change the weight of this feature.
2. DCT Coefficients' variance - This provides a better featural isolation than only DCT Coefficients of a pixel.
3. Intensity Values - Since most of the segments are the same in color(intensities). Color patches are also the features for the segmentation. Hence a pixel is clustered into three color classes.
4. FFT Coefficients - Features obtained at the different points may not be the same but they can be the same in multi-dimensional space. Points with the same FFT coefficient represent similar pattern in the spatial domain of the image.
5. 2-D Gaussian filter - Applied gaussian filter to the image with standard deviation of 0.5 with circular padding. This is one of the fundamental feature clustering should have. For each color filter is applied.
6. Location of a pixel - Generally, it is included in mean-shift segmentation. Hence it is included in k-means.
7. Local variance of intensity - If there are segments with the similar color intensity values then this feature will try to discriminate the regions.

Approach:

1. Every feature is brought down to the scale of 0 to 1. If we want to provide some weight to a feature then weight is multiplied with the feature values.
2. Since DCT coefficients are very robust features, we always include them with weight of 1. Here, for each pixel we calculate DCT coefficients of different colors in the window of 5*5 and take first 10 DCT coefficients because higher coefficients correspond to higher frequencies. The variance of these first 10 coefficient is also important because it completes the whole information of a pixel in that domain.
3. Patches of different colors intensities with size of 5*5 for a pixel are included as features for clustering.
4. Using above features K-means is calculated. K is kept as an input variable from user. 5. Selected pixel will show its segmented region.

Weighing:

1. Features with weight = 1: DCT coefficients, variance of DCT coefficients, Location of a pixel, 2-D Gaussian filter, Local variance of intensity
2. Features with changing weights = FFT coefficients, color intensities.
3. Variability of features - For weighing of color intensities another feature is calculated, which is 2-D Sobel gradient. If Sobel gradient of a pixel is more than the threshold then we decrease the weights of color intensities for that pixel by some factor.
4. If gradient magnitude is more than decided threshold then we increase FFT weight because we want to discriminate a pattern segment.

Implementation

- App takes 3 minutes to compute the segmentation of given image. I have represented segments as shades of gray. Here, we include 141 features for a pixel.

Image of wolf

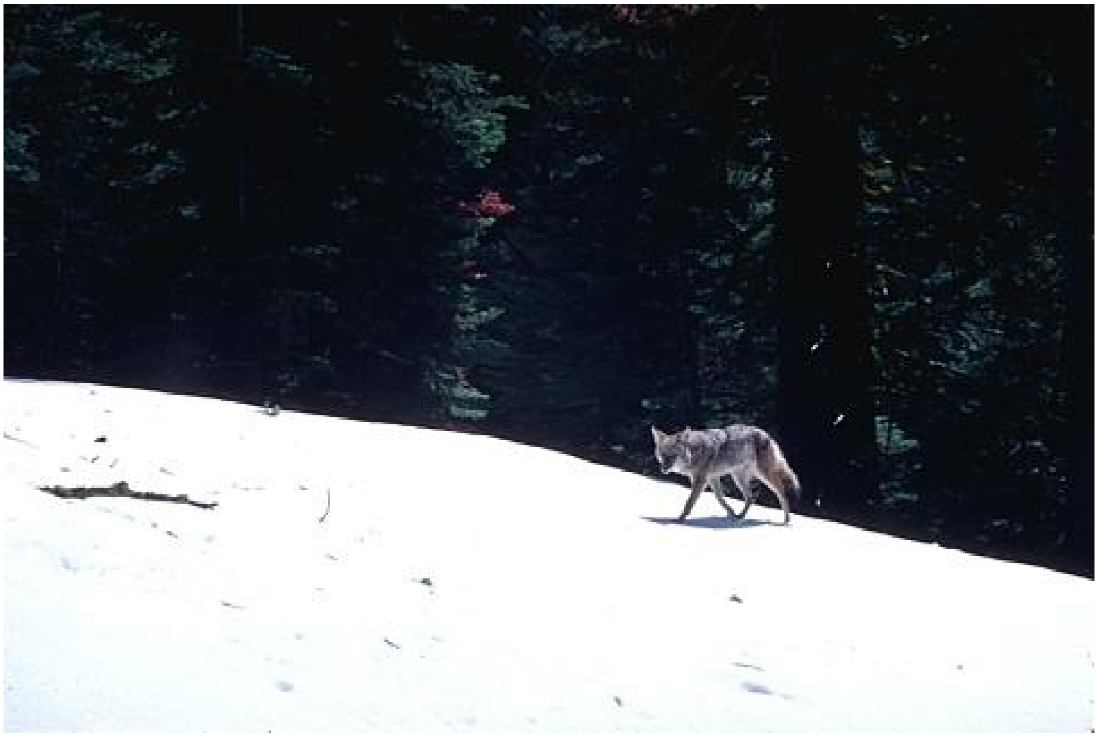


Image of wolf with 3 segments

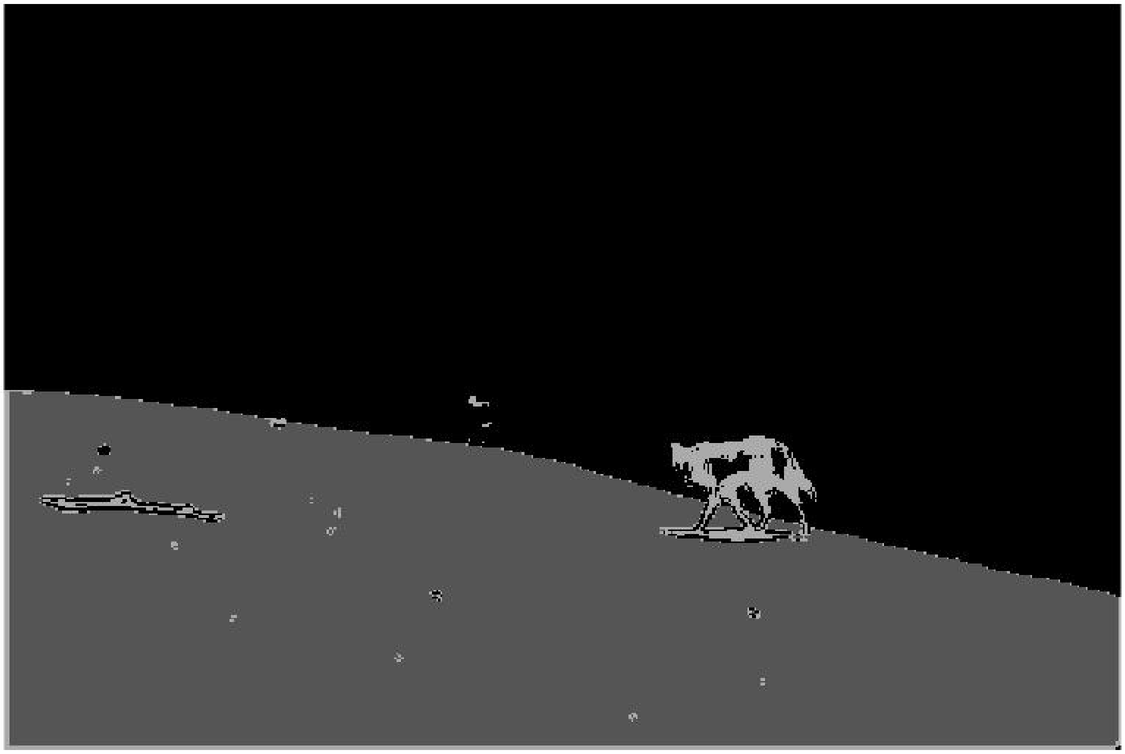


Image of Bird

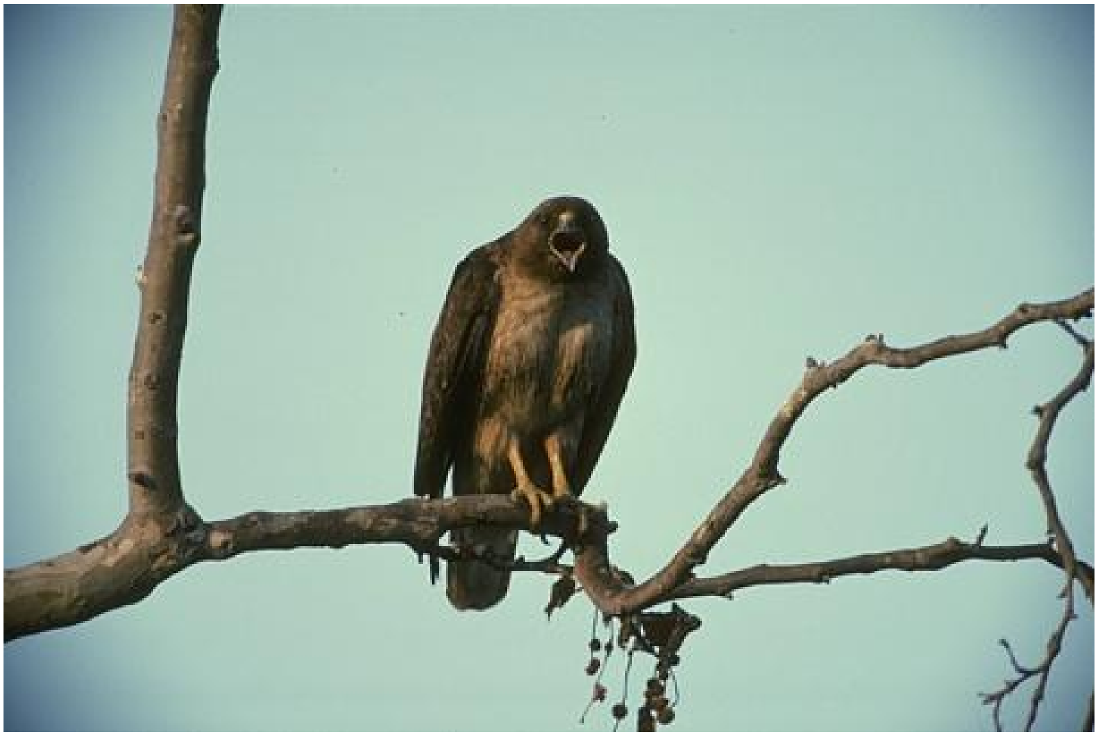


Image of Bird with 2 segments

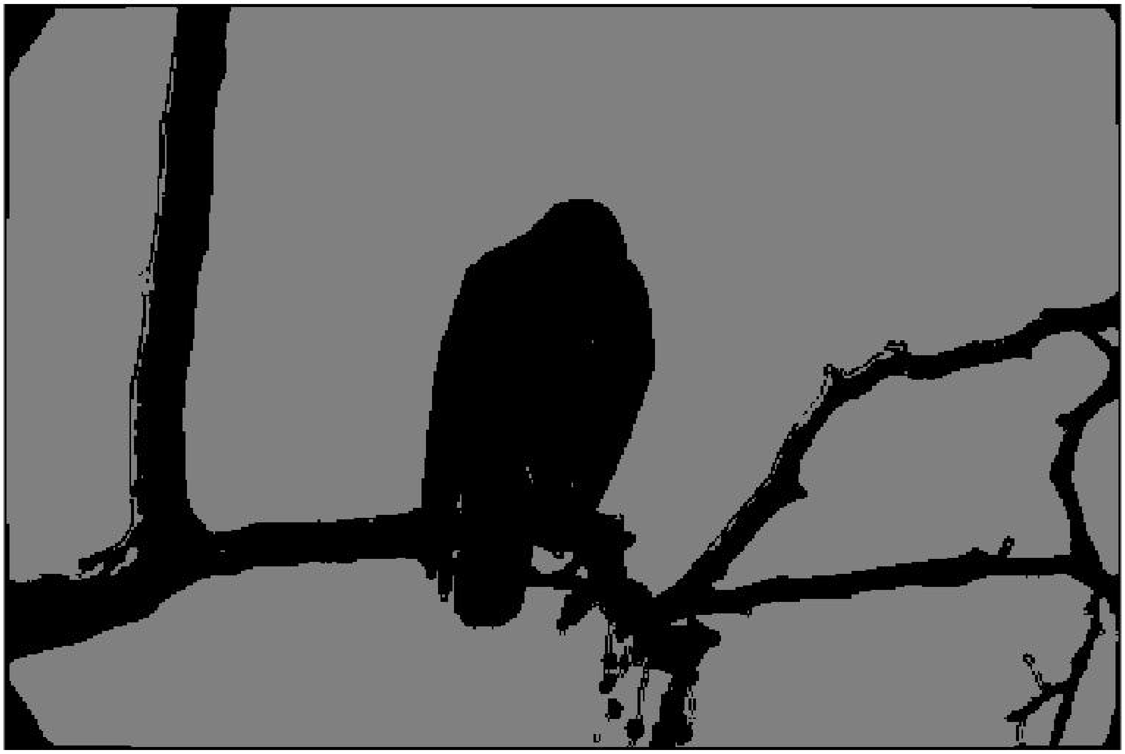


Image of surfer

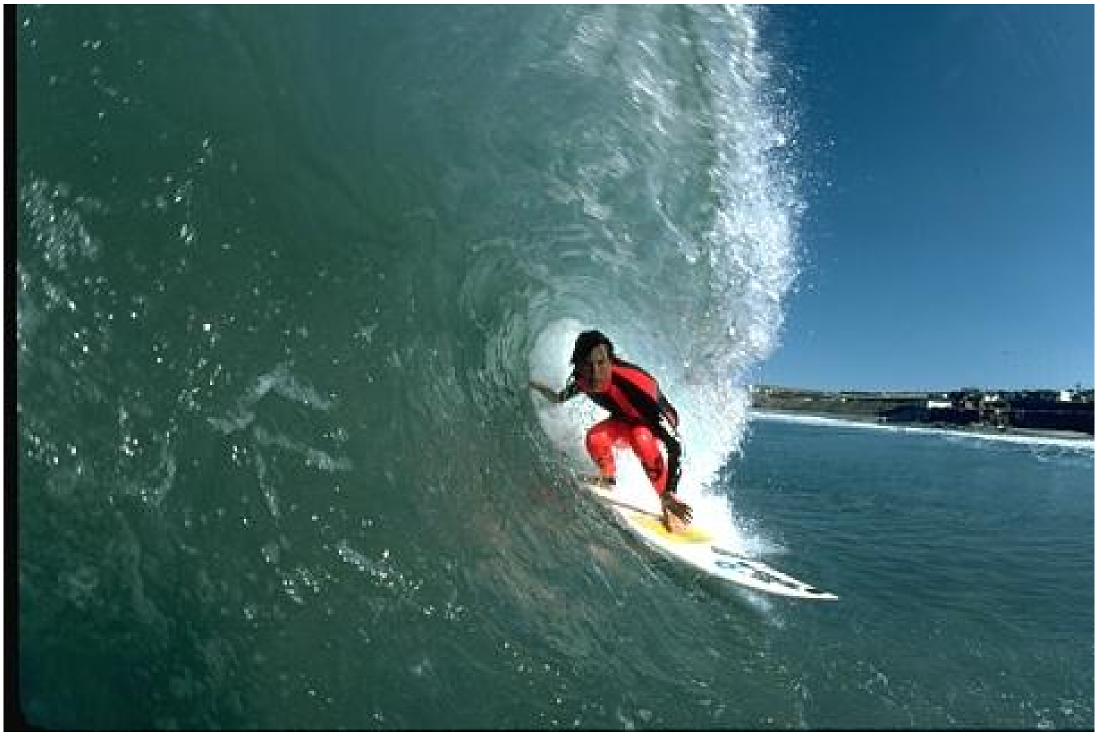


Image of surfer with 4 segments

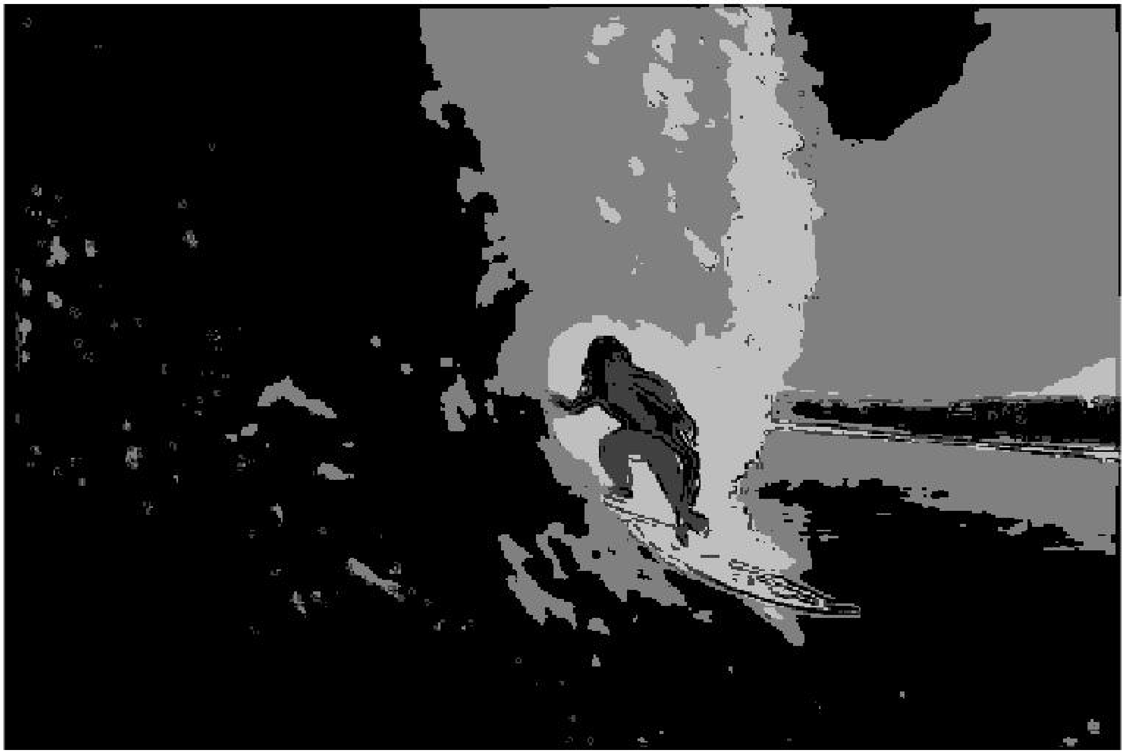
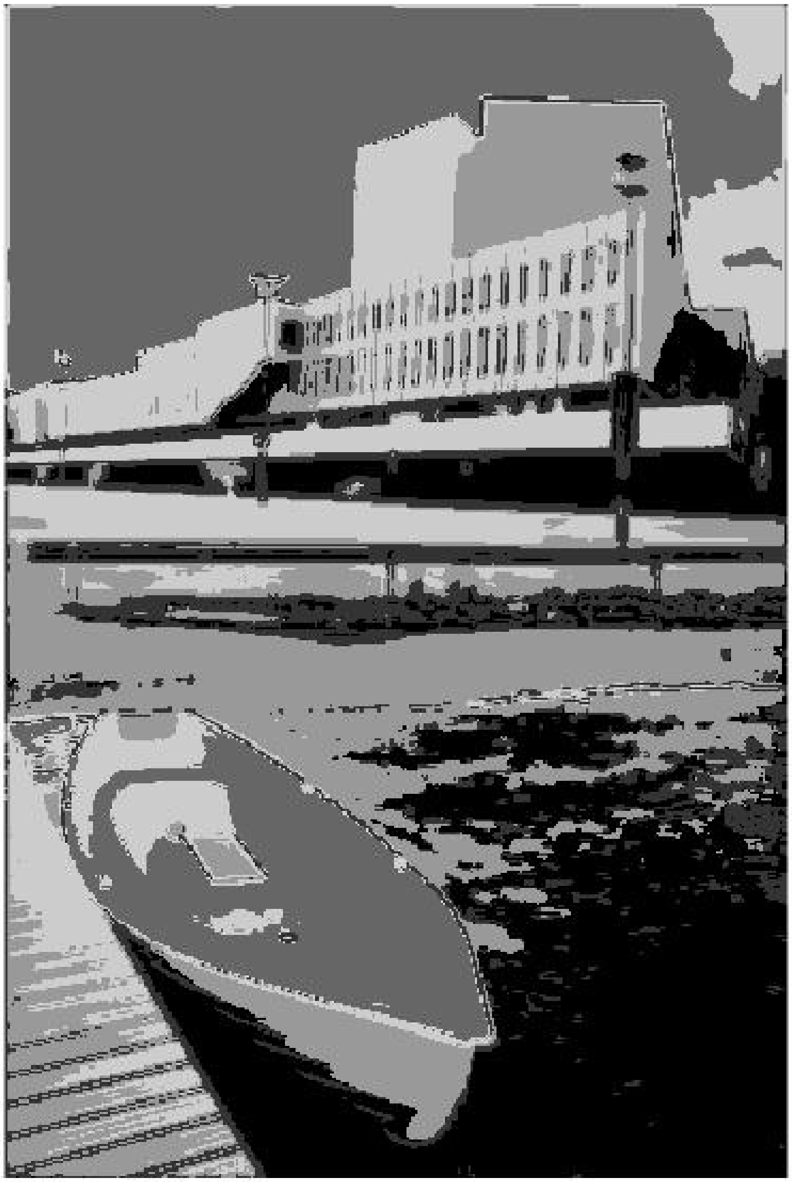


Image of boat

Image of boat with 5 segments



Learning Points

- * How to get an idea of an object in an algorithm? What should we change so that we get targeted region?
* Categories of image segmentation methods
* Typical algorithms of clustering - K-means algorithm, Fuzzy C-Means Algorithm, Gaussian of mixture and mean shift algorithm.
* Maintaining of tracks of large set of data.
* Sometimes, easy and simple looking problems may have very complex solutions.

References:

1. For matlab functions: https://in.mathworks.com/
2. Digital Image Processing, by Gonzalez and Woods
3. Improving Clustering Algorithms for Image Segmentation using Contour and Region Information by Arnau Oliver, Xavier Munoz, Joan Batlle and Jordi Freixenet.
4. Unsupervised Segmentation of Medical Images using DCT Coefficients Pramod K. Singh

Listing 1: Code Snippet: Segmentation

```

startTime = tic;
im = (imread('7.jpg'));
figure(1);
imshow(im);

%prompt = 'Please enter the possible number of segments in the image\n';
k = input(prompt);
%k = 2;

features = zeros(size(im,1), size(im,2),5);
im = double(im);
%variance calculation using 3*3 window
imVar = stdfilt(im);
imVar = imVar/max(max(max(imVar)));
for i=1:3
    features(:,:,i) = imfilter(im(:,:,i),imgaussfilt(im(:,:,i)));
    features(:,:,i) = (features(:,:,i)-min(min(features(:,:,i))))/(max(max(features(:,:,i)))-min(min(features(:,:,i))));
end

% Position as a feature
vec = 1:size(im,2);
features(:,:,4) = repmat(vec,size(im,1),1);
features(:,:,4) = (features(:,:,4)-min(min(features(:,:,4))))/(max(max(features(:,:,4)))-min(min(features(:,:,4))));
vec = 1:size(im,1);
features(:,:,5) = repmat(vec',1,size(im,2));
features(:,:,5) = (features(:,:,5)-min(min(features(:,:,5))))/(max(max(features(:,:,5)))-min(min(features(:,:,5))));

minR = min(min(im(:,:,1)));
minB = min(min(im(:,:,3)));
minG = min(min(im(:,:,2)));
maxR = max(max(im(:,:,1)));
maxG = max(max(im(:,:,2)));
maxB = max(max(im(:,:,3)));

%DCT calculation
psize=2;
DCTFeatures = zeros(size(im,1),size(im,2),10*3+100+6);
im2 = zeros(size(im,1)+4, size(im,2)+4, 3);
im2(psize+1:psize+1+size(im,1)-1, psize+1:psize+1+size(im,2)-1, :) = im;
for i=1:size(im,1)
    for j=1:size(im,2)
        tempDCT1 = dct2(im2(i:i+4 , j:j+4,1));
        tempDCT1 = tempDCT1(:);
        tempDCT1(1:10) = (tempDCT1(1:10)-min(tempDCT1(1:10)))/(max(tempDCT1(1:10))-min(tempDCT1(1:10)));
        colorpatch1 = im2(i:i+4 , j:j+4,1);
        colorpatch1 = (colorpatch1 - minR)/(maxR-minR);
        tempDCT2 = dct2(im2(i:i+4 , j:j+4,2));
        tempDCT2 = tempDCT2(:);
        tempDCT2(1:10) = (tempDCT2(1:10)-min(tempDCT2(1:10)))/(max(tempDCT2(1:10))-min(tempDCT2(1:10)));
        colorpatch2 = im2(i:i+4 , j:j+4,2);
        colorpatch2 = (colorpatch2 - minG)/(maxG-minG);
        tempDCT3 = dct2(im2(i:i+4 , j:j+4,3));
        tempDCT3 = tempDCT3(:);
        tempDCT3(1:10) = (tempDCT3(1:10)-min(tempDCT3(1:10)))/(max(tempDCT3(1:10))-min(tempDCT3(1:10)));
        var1 = var(tempDCT1(1:10));
        var2 = var(tempDCT2(1:10));
        var3 = var(tempDCT3(1:10));
        colorpatch3 = im2(i:i+4 , j:j+4,3);
        colorpatch3 = (colorpatch3 - minB)/(maxB-minB);
        tempFFT1 = abs(fft2(rgb2gray(im2(i:i+4 , j:j+4,:))));
        tempFFT1 = tempFFT1(:);
        tempFFT1(:) = (tempFFT1(:)-min(tempFFT1(:)))/(max(tempFFT1(:))-min(tempFFT1(:)));
        DCTFeatures(i,j,:) = cat(1,tempDCT1(1:10),tempDCT2(1:10),tempDCT3(1:10),tempFFT1(:),colorpatch1(:),colorpatch2(:),colorpatch3(:),var1,var2,var3,imVar(i,j,3),imVar(i,j,2),imVar(i,j,1));
    end
end

%weighting
im_sobelR = GSobel(im(:,:,1),3);
im_sobelG = GSobel(im(:,:,2),3);
im_sobelB = GSobel(im(:,:,3),3);

for i=1:size(im,1)
    for j=1:size(im,2)
        if im_sobelG(i,j)/max(max(im_sobelG))>0.5
            DCTFeatures(i,j,81:105) = 0.01*DCTFeatures(i,j,81:105);
        end
        if im_sobelR(i,j)/max(max(im_sobelR))>0.5
            DCTFeatures(i,j,56:80) = 0.01*DCTFeatures(i,j,56:80);
        end
        if im_sobelB(i,j)/max(max(im_sobelB))>0.3
            DCTFeatures(i,j,106:130) = 0.01*DCTFeatures(i,j,106:130);
        end
        if imVar(i,j,:) <0.1
            DCTFeatures(i,j,31:55) = 0.01 * DCTFeatures(i,j,31:55);
        end
    end
end

% concatenation
features = cat(3,features,DCTFeatures);
```

```
reshapedFeatures = reshape(features, [size(features,1)*size(features,2) size(features,3)]);

%%clustering
ids = kmeans(reshapedFeatures,k);
reshapedIdx = reshape(ids, [size(im,1) size(im,2)]);

figure(2);
im_segmented = im(:,:,:);

% shades of gray
 for i = 1:size(im,1)
     for j = 1:size(im,2)
         im_segmented(i,j,:) = [(reshapedIdx(i,j)-1)*255.0/(k),(reshapedIdx(i,j)-1)*255.0/(k),(reshapedIdx(i,j)-1)*255.0/(k)];
     end
 end

%for i=1:k
%    imshow(reshapedIdx == i);
%    waitforbuttonpress;
%end

imshow(uint8(im_segmented));
%figure(3);
%imshow(imoverlay(im,reshapedIdx,[1 1 1]));
elapsedTime = toc(startTime);
message = sprintf('Done!\n\nElapsed time = %.2f seconds.', elapsedTime);
disp(message);
```

Listing 2: Code Snippet: Generalized Sobel

```
function [E Mx My]= GSobel(img,Wsize)
for i=1:Wsize
    Sx(i)=factorial((Wsize-1))/((factorial((Wsize-1)-(i-1)))*(factorial(i-1)));
    Dx(i)=Pasc(i-1,Wsize-2)-Pasc(i-2,Wsize-2);
end
Sy=Sx';
Dy=Dx';
Mx=Sy(:)*Dx;
My=Mx';
Ey=imfilter(double(img),My,'symmetric');
Ex=imfilter(double(img),Mx,'symmetric');
E=sqrt(Ex.^2+Ey.^2);
%figure,imshow(img,[]),title('Original Image');
%figure,imshow(E,[]),title('Edge Image');

function P=Pasc(k,n)
if (k>=0)&&(k<=n)
    P=factorial(n)/(factorial(n-k)*factorial(k));
else
    P=0;
end
```