CSE166 Assignment #:
Due Date:

Please list all contributors to the assignment below.
Staple this sheet to the front of your submission.

1 Student Name(s):

1. _____
2. _____
3. _____

2 Student ID(s):

1. _____
2. _____
3. _____

3 Email Address(es):

1. _____
2. _____
3. _____

4 Other References (optional):

CSE166 Homework 4

Nitay Joffe

10/24/2006

Written exercises

1. *GW, Problem 6.4.*
   We create three color filters that are closely related to the wavelengths of the colors of the three objects. For each filter, only the colors that resemble it closely will produce a high effect in the camera, while others will have low ones. A filter wheel can be put in to control the position of the filter at all times. The color white responses to all the three filters equally with high values, while the color black responds equally to al three filters with low values.

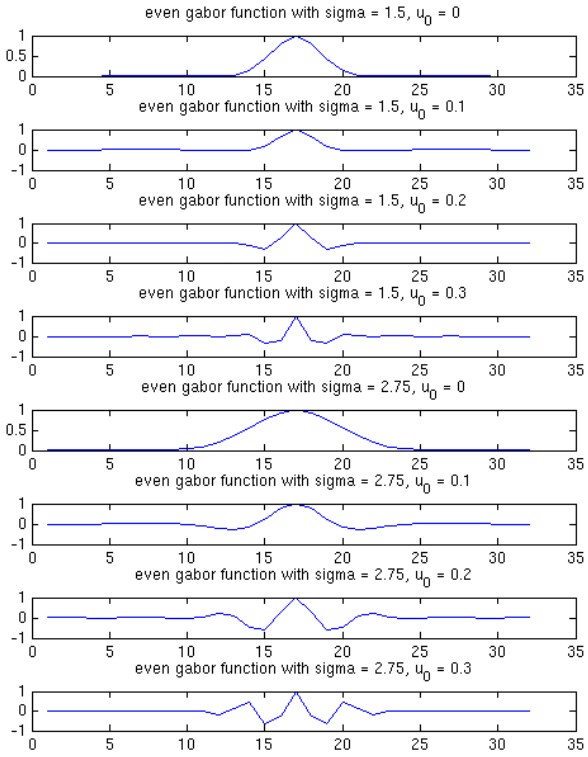
2. *GW, Problem 6.5.*
   R = 0.5, G = 1.0, B = 0.5. The color is teal.

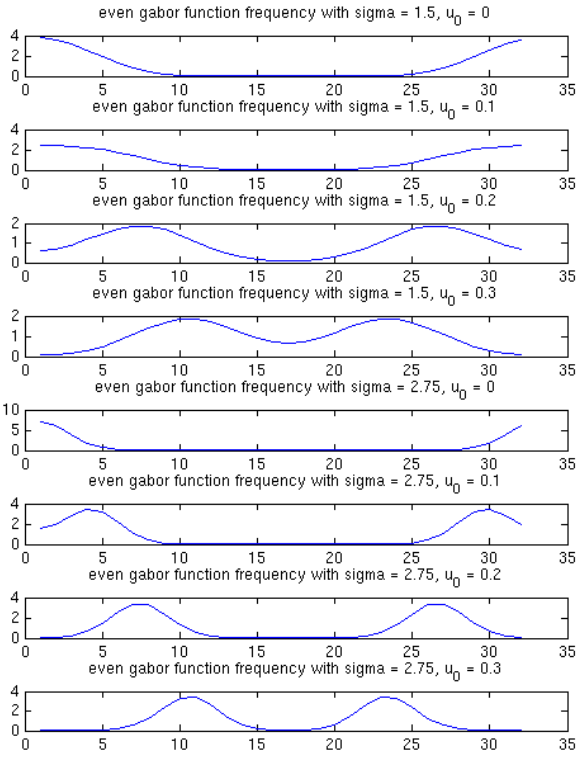
3. *GW, Problem 6.6.*

4. *GW, Problem 6.7.*
   Gray occurs when R = G = B, so there are 2^8, or 256, shades of gray.

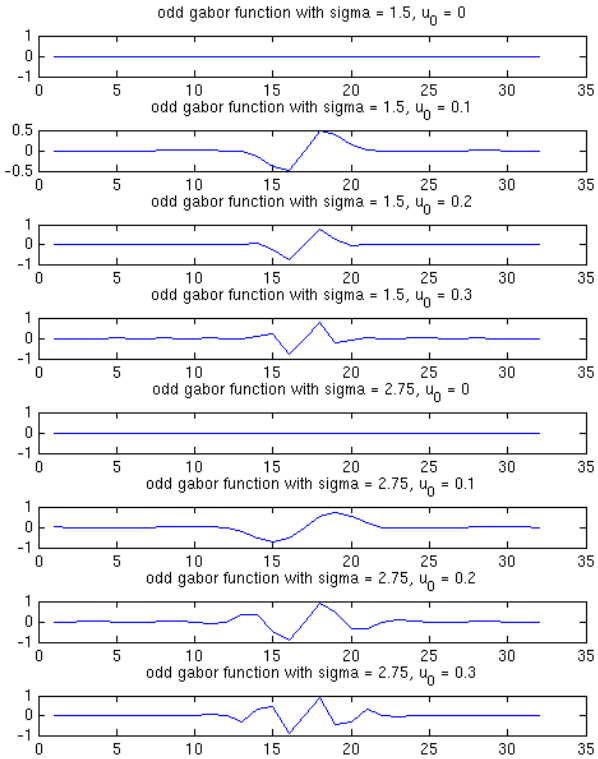
5. *GW, Problem 6.12.*

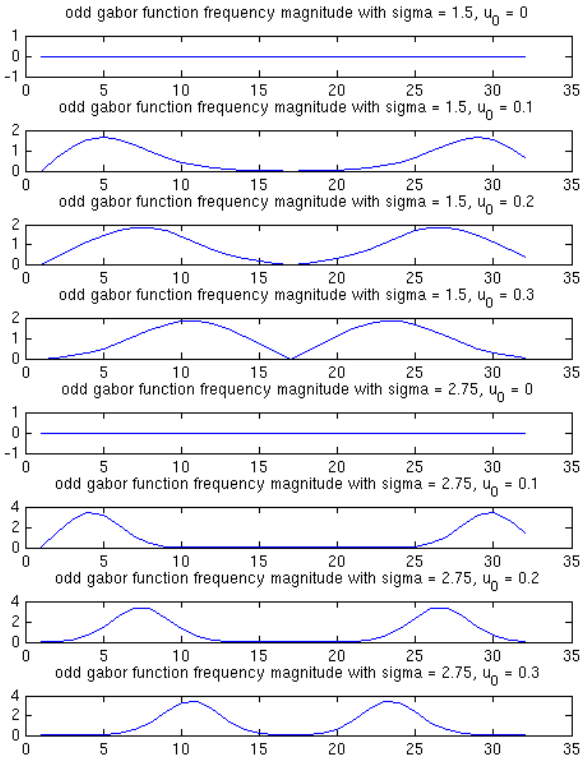
1a – 1D Gabor even functions and their frequencies





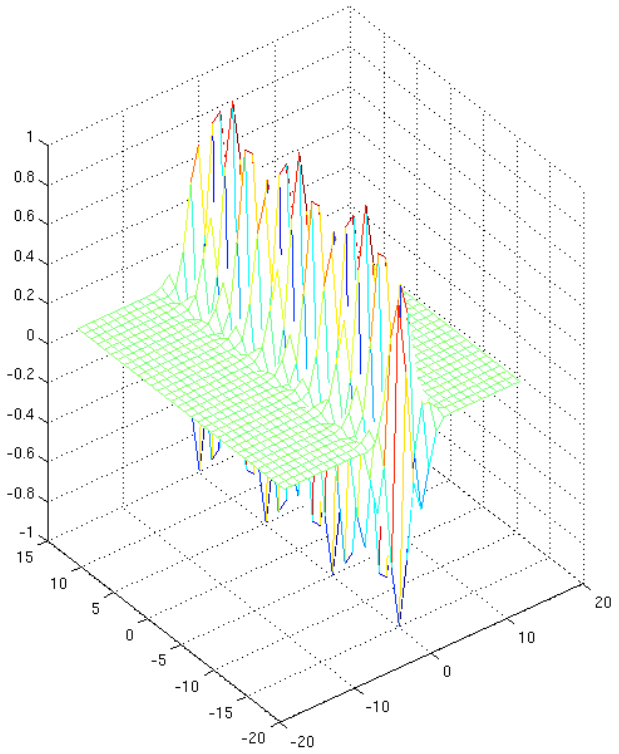
1a – 1D Gabor odd functions and their frequencies



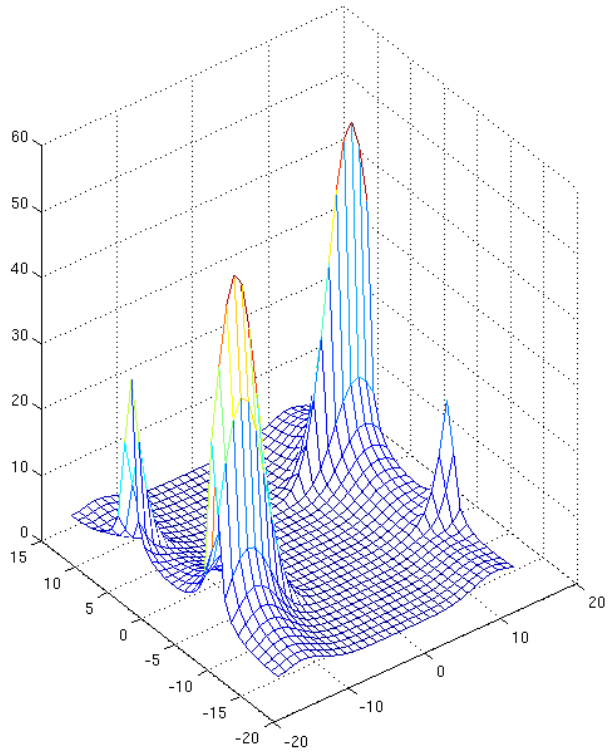


1b – 2D Gabor functions and their frequencies

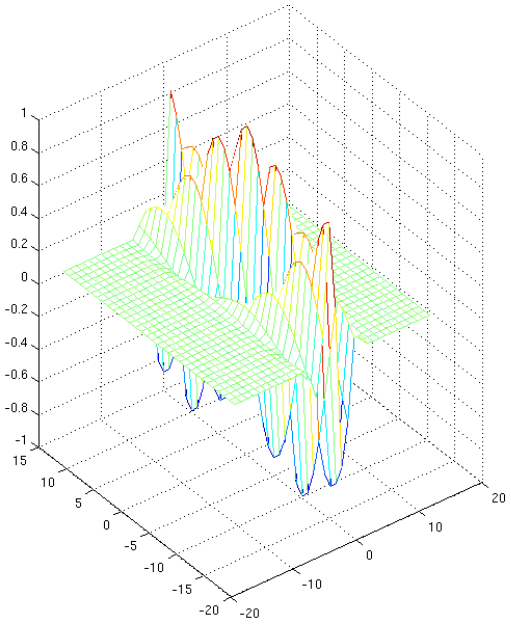
even gabor function with $\sigma = 1.75$, $u_0 = 0.1$, $v_0 = 0.3$



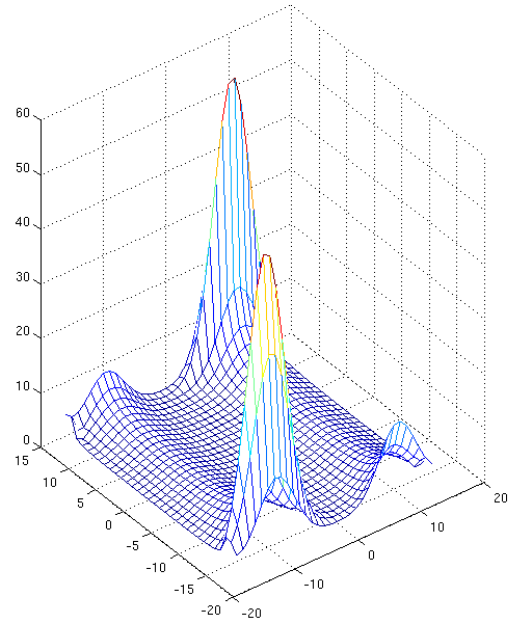
even gabor function frequency with $\sigma = 1.75$, $u_0 = 0.1$, $v_0 = 0.3$



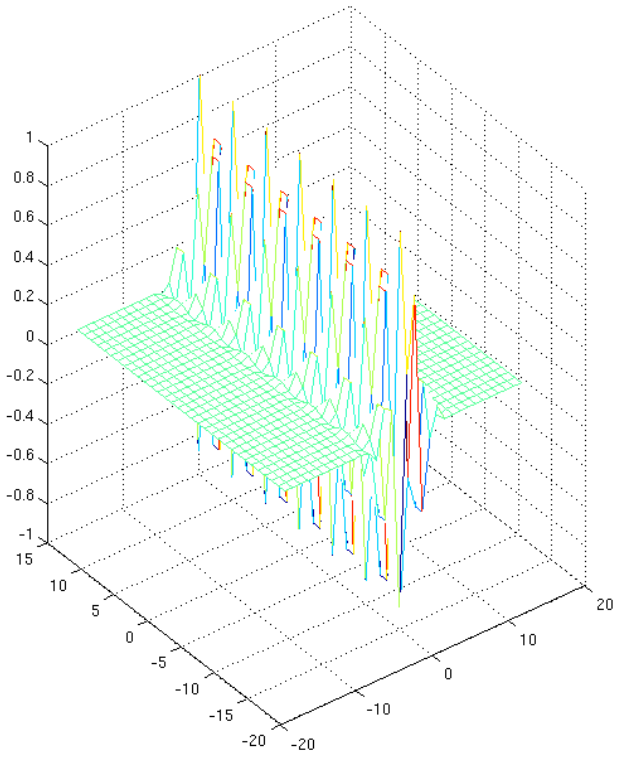
even gabor function with $\sigma = 1.75$, $u_0 = 0.2$, $v_0 = 0.05$



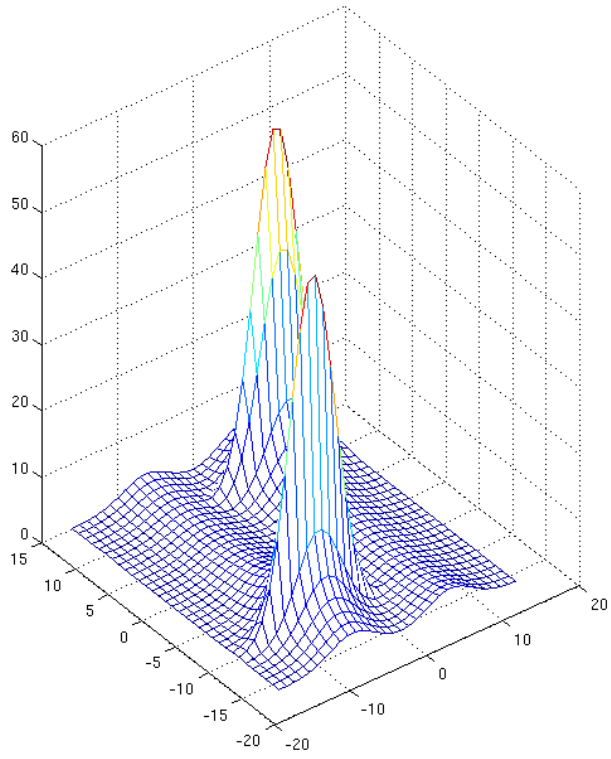
even gabor function frequency with $\sigma = 1.75$, $u_0 = 0.2$, $v_0 = 0.05$



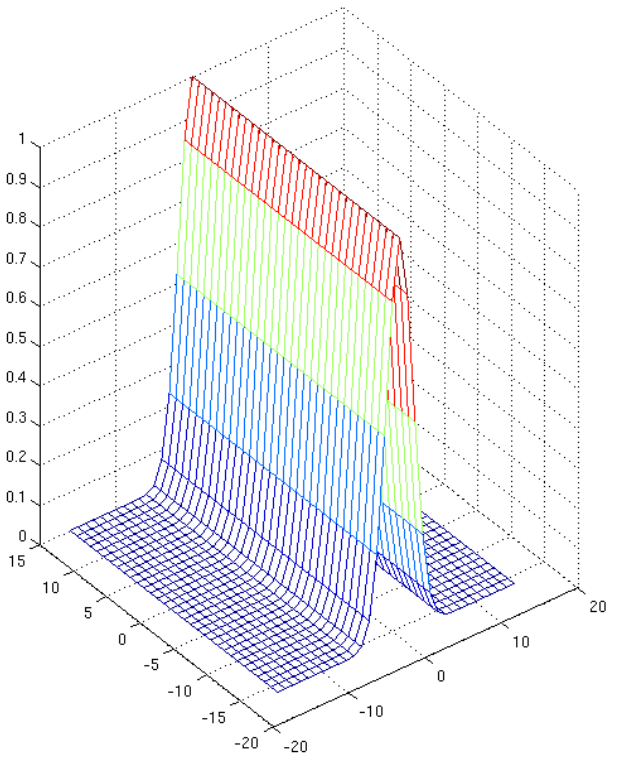
even gabor function with $\sigma = 1.75$, $u_0 = 0.3$, $v_0 = 0.2$



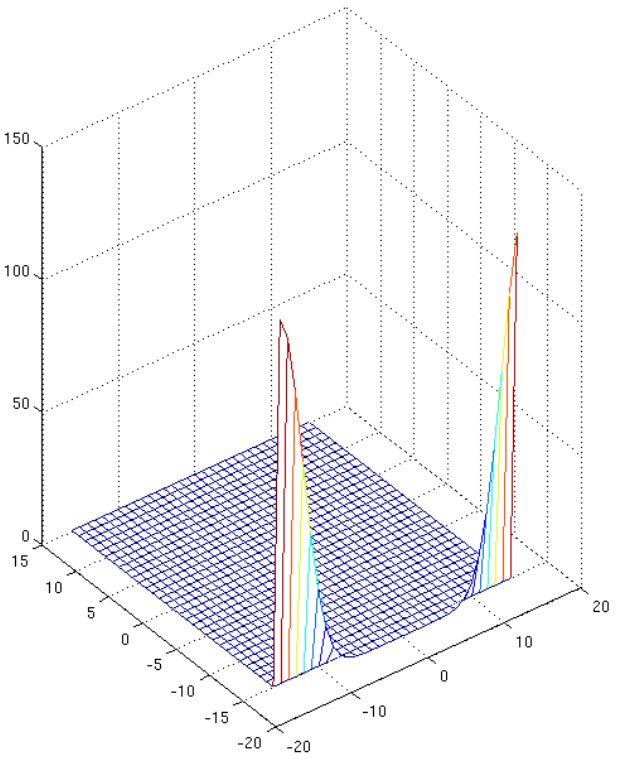
even gabor function frequency with $\sigma = 1.75$, $u_0 = 0.3$, $v_0 = 0.2$



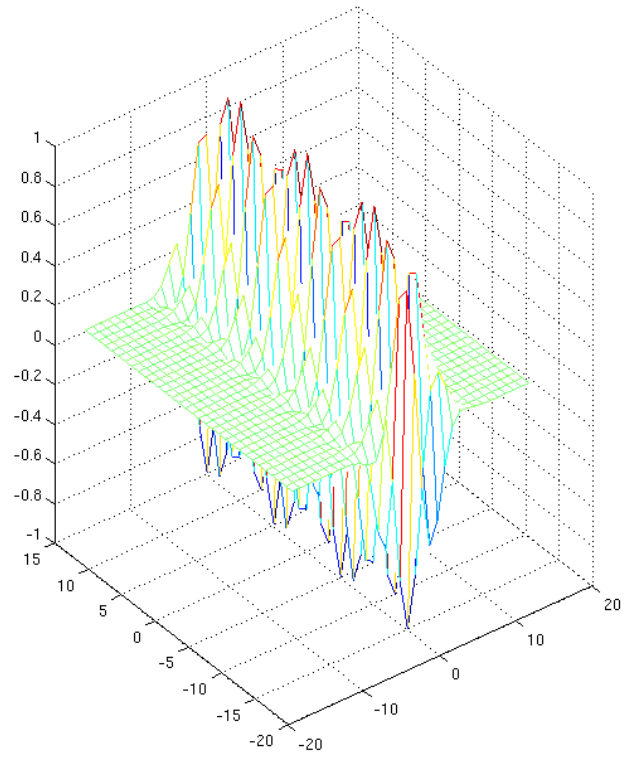
even gabor function with $\sigma = 1.75$, $u_0 = 0$, $v_0 = 0$



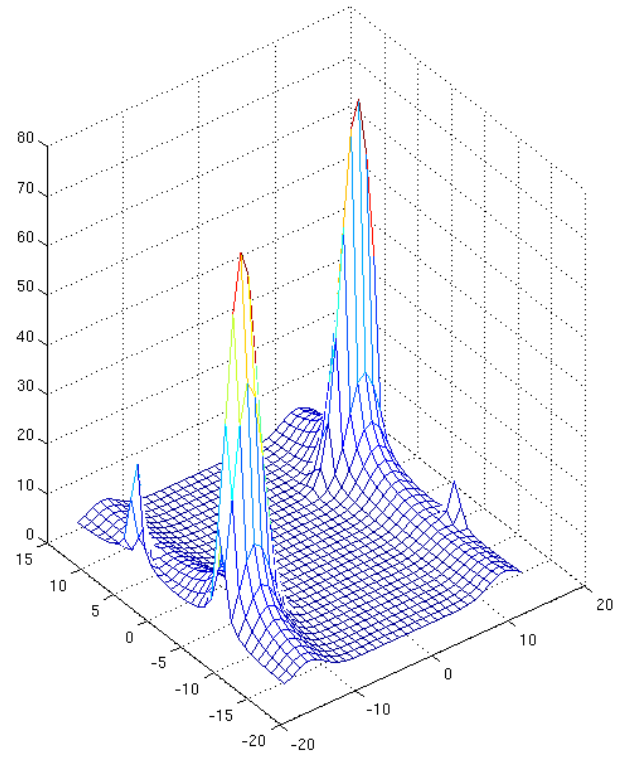
even gabor function frequency with $\sigma = 1.75$, $u_0 = 0$, $v_0 = 0$



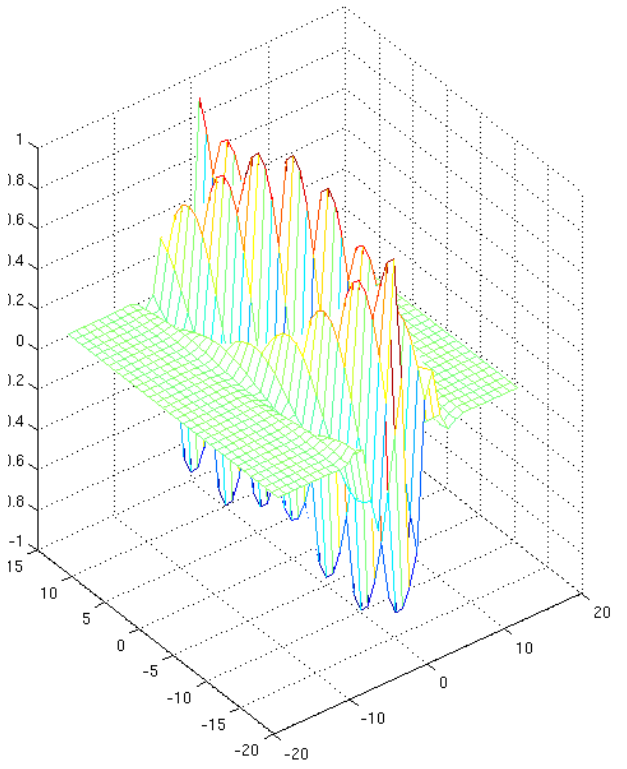
even gabor function with $\sigma = 2.5$, $u_0 = 0.1$, $v_0 = 0.3$



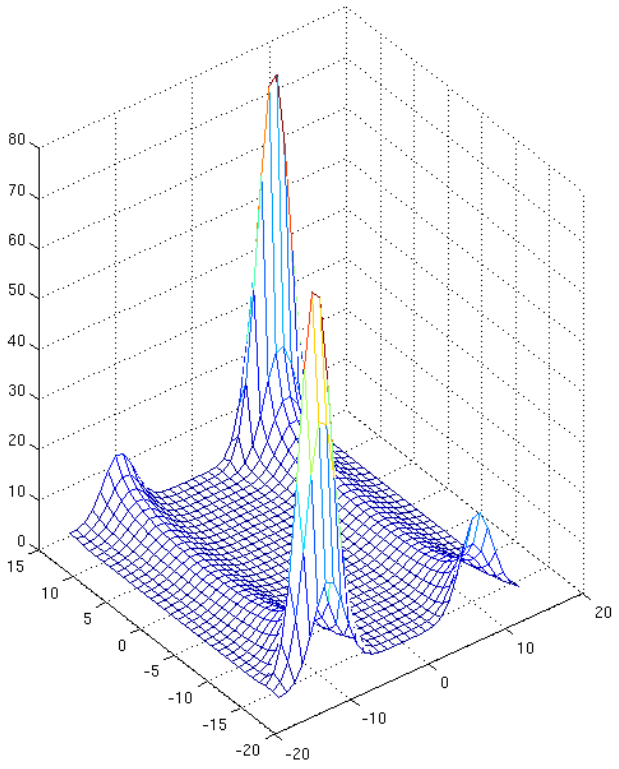
even gabor function frequency with $\sigma = 2.5$, $u_0 = 0.1$, $v_0 = 0.3$



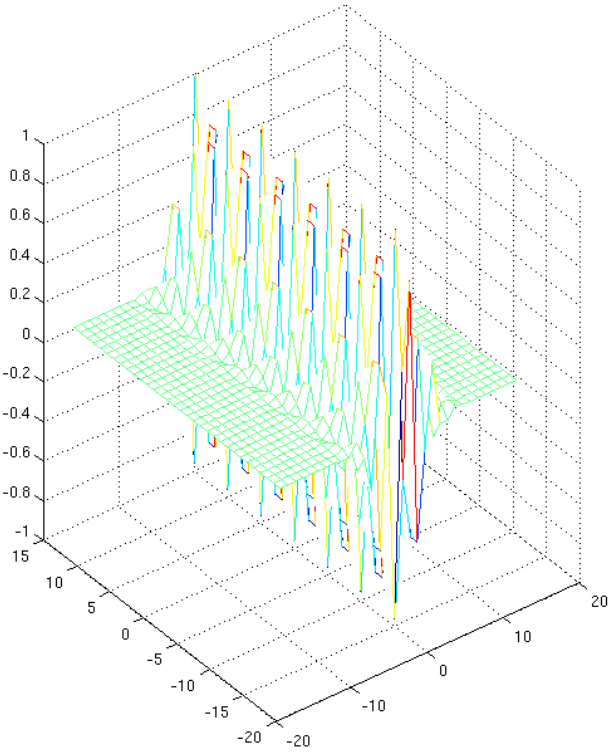
even gabor function with $\sigma = 2.5$, $u_0 = 0.2$, $v_0 = 0.05$



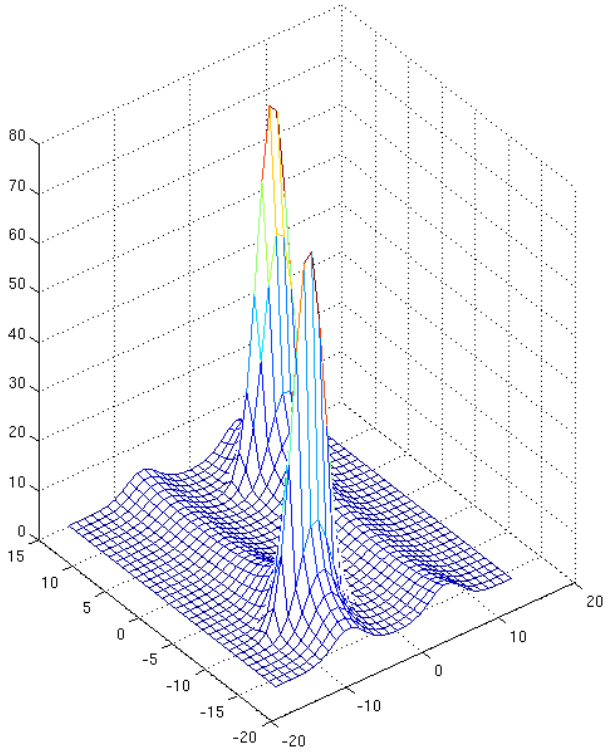
even gabor function frequency with $\sigma = 2.5$, $u_0 = 0.2$, $v_0 = 0.05$



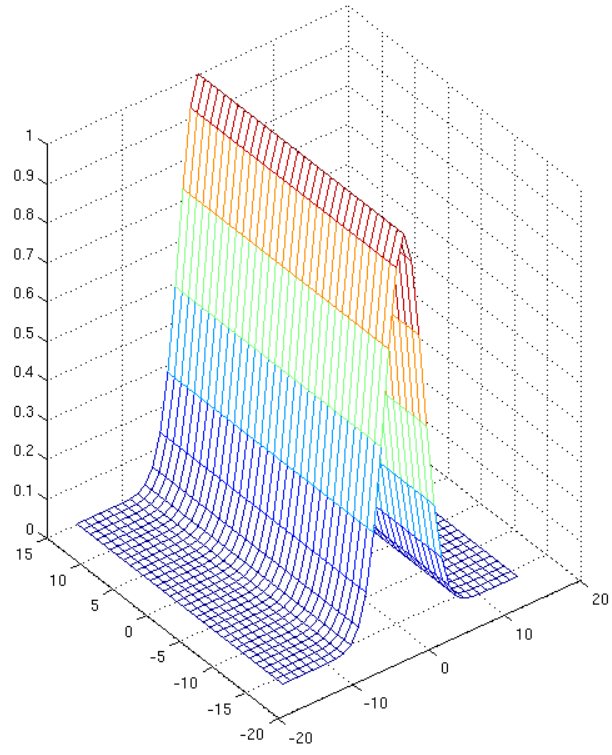
even gabor function with $\sigma = 2.5$, $u_0 = 0.3$, $v_0 = 0.2$



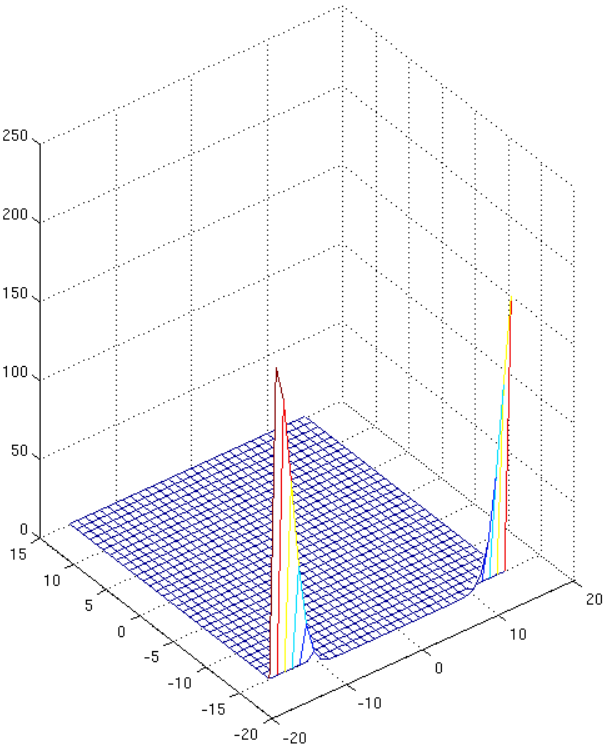
even gabor function frequency with $\sigma = 2.5$, $u_0 = 0.3$, $v_0 = 0.2$



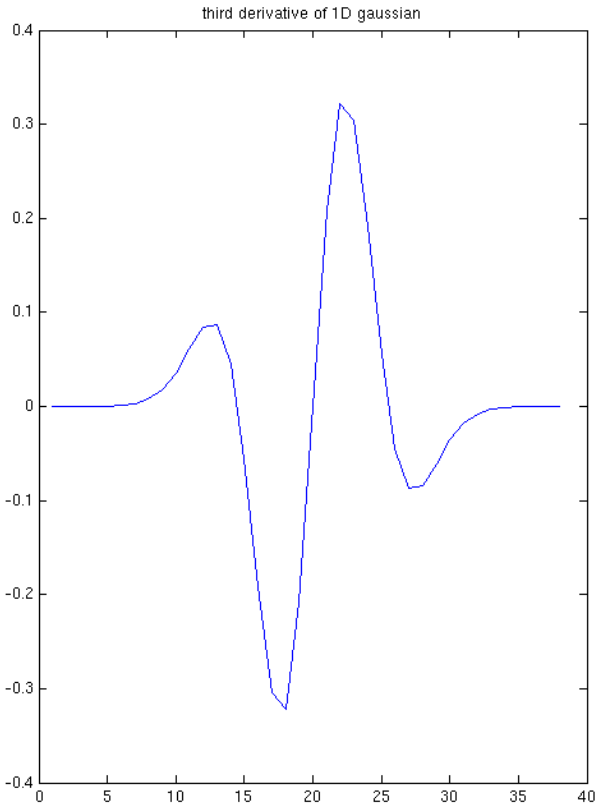
even gabor function with $\sigma = 2.5$, $u_0 = 0$, $v_0 = 0$

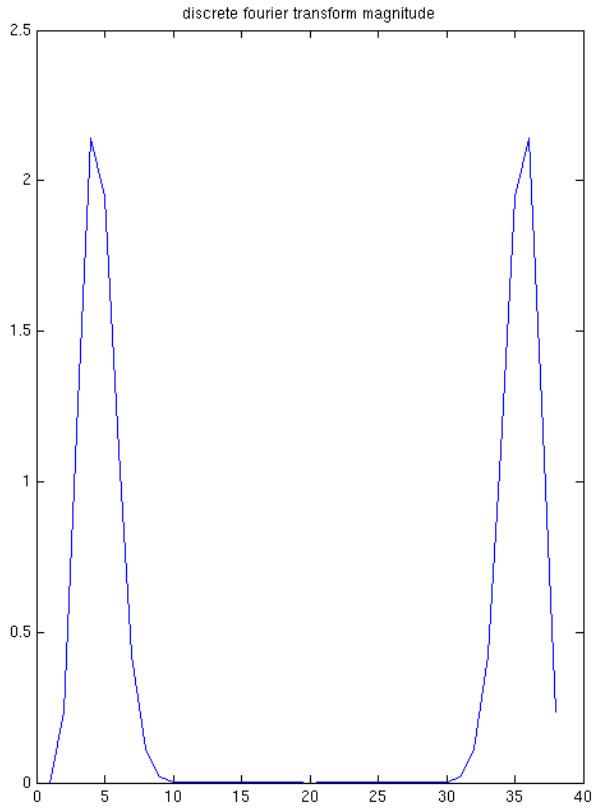


even gabor function frequency with $\sigma = 2.5$, $u_0 = 0$, $v_0 = 0$

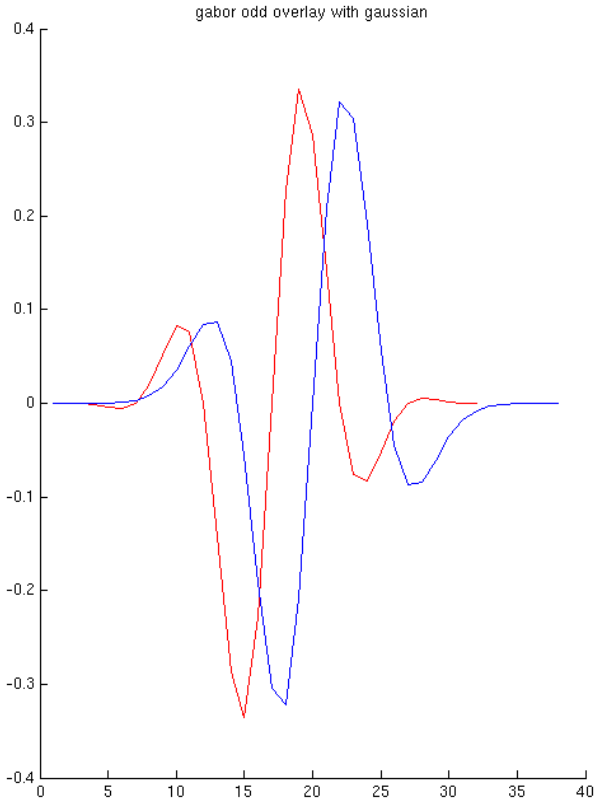


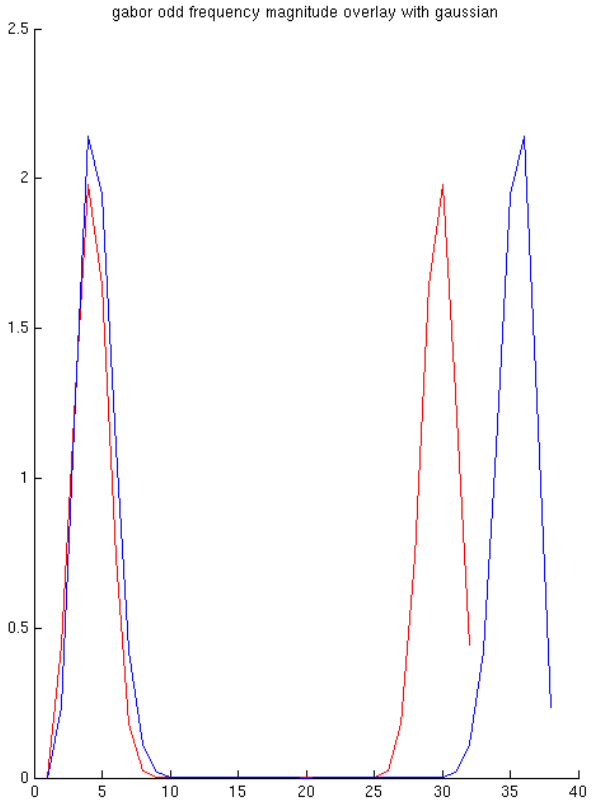
2a – 1D Gaussian Third Derivative



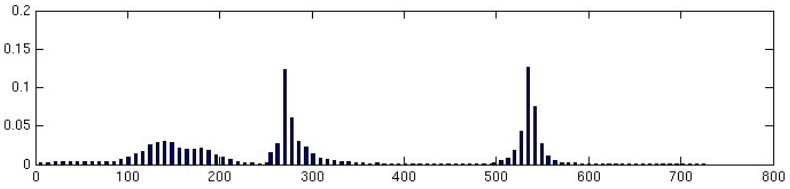


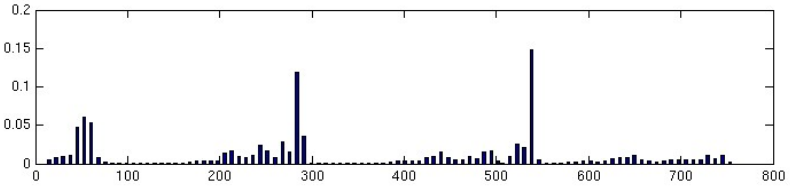
2b – Approximation of Gaussian with Gabor

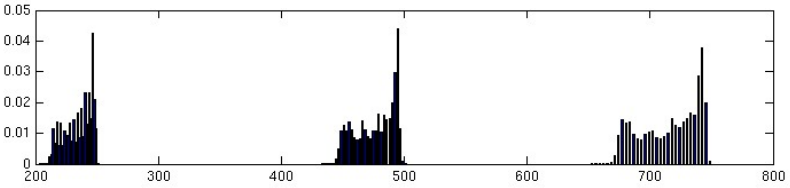


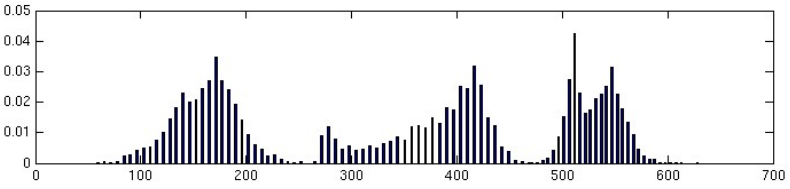


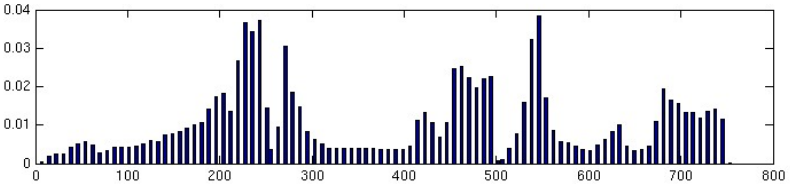
3b – Color Histograms

3c – Chi Squared Distance

0	0.5291	0.6130	0.4293	0.3900
0.5291	0	0.5313	0.6083	0.3101
0.6130	0.5313	0	0.2852	0.1965
0.4293	0.6083	0.2852	NaN	0.3755
0.3900	0.3101	0.1965	0.3755	0

```
% Author: <njoffe@ucsd.edu> Nitay Joffe
% Date: 10/24/2006
% Class: CSE 166 - Image Processing
% Homework: 1
% Problem: 1 - Gabor Functions
clear;

% 1. Gabor Functions
%   In 1D, the Gabor function of width sigma and spatial frequency u_o is given
%   by the following complex-valued function:
%       h(x) = e^{-x^2 / 2\sigma^2} e^{j2\pi u_o x}
%   In 2D, the corresponding expression is
%       h(x) = e^{-||X||^2 / 2\sigma^2} e^{j2\pi U_o X}
%   where X = (x, y) and U_o = (u_o , v_o), and it serves as an oriented
%   bandpass filter. The even and odd Gabor functions are equal to the real and
%   imaginary parts of h, respectively.

% (a) Compute eight examples of even and/or odd 1D Gabor functions on the
%     interval x: [-16, 15] using parameters chosen in the following ranges:
%     sigma: [1, 3] and u_o: [0, 0.3]. For each example, plot the function and
%     its Fourier transform magnitude.
x = -16:15;
sigma_values = [1.5 2.75];
u_0_values = [0.0 0.1 0.2 0.3];
number_of_items = numel(sigma_values) * numel(u_0_values);

plot_even_function = true;
i = 1;

figure;
for sigma = sigma_values
  for u_0 = u_0_values
    gabor_function = exp(-x.*x/(2*sigma^2)).*exp(j*2*pi*u_0*x);
    even_gabor_function = real(gabor_function);
    odd_gabor_function = imag(gabor_function);

    gabor_function_frequency = fft(gabor_function);
    even_gabor_function_frequency_magnitude = abs(fft(even_gabor_function));
    odd_gabor_function_frequency_magnitude = abs(fft(odd_gabor_function));

    subplot(number_of_items,2,i);
    if plot_even_function
      plot(even_gabor_function);
      title(['even gabor function with sigma = ' num2str(sigma) ', u_0 = ' num2str(u_0)]);
    else
      plot(odd_gabor_function);
      title(['odd gabor function with sigma = ' num2str(sigma) ', u_0 = ' num2str(u_0)]);
    end

    subplot(number_of_items,2,i+1);
    if plot_even_function
      plot(even_gabor_function_frequency_magnitude);
      title(['even gabor function frequency with sigma = ' num2str(sigma) ', u_0 = ' num2str(u_0)]);
    else
      plot(odd_gabor_function_frequency_magnitude);
      title(['odd gabor function frequency magnitude with sigma = ' num2str(sigma) ', u_0 = ' num2str(u_0)]);
    end
    i = i + 2;
  end
end

% (b) Compute eight examples of even and/or odd 2D Gabor functions on the
%     interval x: [-16, 15]x[-16, 15] using parameters chosen in the following
%     ranges: \sigma: [1, 3] and u_o = [0, 0.3]x[0, 0.3]. For each example,
%     display the function (either as an image or a surface plot) and its
%     Fourier transform magnitude.
[x,y] = meshgrid(-16:15,-16:15);
sigma_values = [1.75 2.5];
u_0_values = [0 0; 0.1 0.3; 0.3 0.2; 0.2 0.05];
number_of_u_0_rows = size(u_0_values,1);

plot_even_function = true;
```

```
for sigma = sigma_values
  for u_0_row_index = 1:number_of_u_0_rows
    u_0 = u_0_values(u_0_row_index,1);
    v_0 = u_0_values(u_0_row_index,2);

    u_0_x_dot_product = u_0 * x + v_0 * y;
    gabor_function = exp(-abs(x.*x)/(2*sigma^2)).*exp(j*2*pi*u_0_x_dot_product);
    even_gabor_function = real(gabor_function);
    odd_gabor_function = imag(gabor_function);

    gabor_function_frequency = fft2(gabor_function);
    even_gabor_function_frequency_magnitude = abs(fft2(even_gabor_function));
    odd_gabor_function_frequency_magnitude = abs(fft2(odd_gabor_function));

    figure;
    subplot(1,2,1);
    if plot_even_function
      mesh(x,y,even_gabor_function);
      title(['even gabor function with \sigma = ' num2str(sigma) ', u_0 = ' num2str(u_0), ', v_0 = '
num2str(v_0)]);
    else
      mesh(x,y,odd_gabor_function);
      title(['odd gabor function with \sigma = ' num2str(sigma) ', u_0 = ' num2str(u_0), ', v_0 = '
num2str(v_0)]);
    end

    subplot(1,2,2);
    if plot_even_function
      mesh(x,y,even_gabor_function_frequency_magnitude);
      title(['even gabor function frequency with \sigma = ' num2str(sigma) ', u_0 = ' num2str(u_0), ', v_0 = '
num2str(v_0)]);
    else
      mesh(x,y,odd_gabor_function_frequency_magnitude);
      title(['odd gabor function frequency magnitude with \sigma = ' num2str(sigma) ', u_0 = ' num2str(u_0),
', v_0 = ' num2str(v_0)]);
    end
  end
end
```

```
% Author: <njoffe@ucsd.edu> Nitay Joffe
% Date: 10/24/2006
% Class: CSE 166 - Image Processing
% Homework: 1
% Problem: 2 - Gaussian derivatives vs. Gabor functions
clear;

% Gaussian derivatives vs. Gabor functions
% (a) Evaluate the 1D Gaussian function e^{-x^2/2\sigma^2} on the interval
%     x: [-16, 15] using \sigma = 3. Compute its approximate third derivative
%     by repeated application of the centered first difference kernel. Plot the
%     resulting function and its DFT magnitude.

x = -16:15;
sigma = 3;
y = exp(-x.^2/(2*sigma^2));
centered_first_difference_kernel = [1 0 -1];

first_derivative = conv(centered_first_difference_kernel, y);
second_derivative = conv(centered_first_difference_kernel, first_derivative);
third_derivative = conv(centered_first_difference_kernel, second_derivative);
third_derivative_frequency_magnitude = abs(fft(third_derivative));

figure;
subplot(1,2,1);
plot(third_derivative);
title('third derivative of 1D gaussian');
subplot(1,2,2);
plot(third_derivative_frequency_magnitude);
title('discrete fourier transform magnitude');

% (b) Experimentally find parameter values for an even or odd 1D Gabor function
%     that closely matches the Gaussian 3rd derivative. Note: you may also need
%     to apply an amplitude scaling factor. Once you have found a good match,
%     make plots showing the quality of the match in both the spatial domain and
%     the frequency domain.

x = -16:15;
sigma = 4;
u_o = 1.1;
amplitude_scaling_factor = 0.4;

gabor = amplitude_scaling_factor * exp(-x.^2 / (2*sigma^2)).*exp(j*2*pi*u_o*x);
gabor_odd = imag(gabor);
gabor_odd_frequency_magnitude = abs(fft(gabor_odd));

figure;
subplot(1,2,1);
hold on;
plot(gabor_odd,'r');
plot(third_derivative);
title('gabor odd overlay with gaussian');
subplot(1,2,2);
hold on;
plot(gabor_odd_frequency_magnitude,'r');
plot(third_derivative_frequency_magnitude);
title('gabor odd frequency magnitude overlay with gaussian');
```

```
% Author: <njoffe@ucsd.edu> Nitay Joffe
% Date: 10/24/2006
% Class: CSE 166 - Image Processing
% Homework: 1
% Problem: 3 - Color Histograms
% Question: a

% (a) Make a function in Matlab that computes the RGB color histogram for a
%     user-selected rectangular region in an arbitrary color image. Use 32 bins
%     for each color channel, spaced equally between 0 and 255. Concatenate the
%     three histograms together to make a combined histogram of length
%     3x32 = 96. Hint: to get the histogram for the red channel,
%     use rh=hist(R(:),32), where R is an array containing the red slice of an
%     RGB image. Once you have computed the combined histogram, normalize it so
%     that it sums to 1.
function [rgb_image_rectangle,combined_histogram,combined_position] = rgb_histogram(rgb_image)
  number_of_bins = 32;

  % We assume imshow(rgb_image) has already been displayed by the caller.
  rgb_image_rectangle = getrect;
  xmin = rgb_image_rectangle(1);
  ymin = rgb_image_rectangle(2);
  xmax = xmin + rgb_image_rectangle(3);
  ymax = ymin + rgb_image_rectangle(4);

  rgb_image_rectangle = rgb_image(ymin:ymax,xmin:xmax,:);

  red_layer_double = double(rgb_image_rectangle(:,:,1));
  [red_histogram,red_position] = hist(red_layer_double(:), number_of_bins);

  green_layer_double = double(rgb_image_rectangle(:,:,2));
  [green_histogram,green_position] = hist(green_layer_double(:), number_of_bins);
  green_position = green_position + red_position(end);

  blue_layer_double = double(rgb_image_rectangle(:,:,3));
  [blue_histogram,blue_position] = hist(blue_layer_double(:), number_of_bins);
  blue_position = blue_position + green_position(end);

  combined_histogram = [red_histogram green_histogram blue_histogram];
  combined_histogram = combined_histogram ./ sum(combined_histogram);

  combined_position = [red_position green_position blue_position];
end
```

```
% Author: <njoffe@ucsd.edu> Nitay Joffe
% Date: 10/24/2006
% Class: CSE 166 - Image Processing
% Homework: 1
% Problem: 3 - Color Histograms
% Question: c

% (c) Implement a function to compute the \chi^2 distance between a pair of
%     normalized histograms h_i(k) and h_j(k), which is defined as:
%
%     \chi^2(i,j) =
%         \frac{1}{2}\sum_{k=1}^{K}\frac{[h_i(k)-h_j(k)]^2}{h_i(k)+h_j(k)}
function distance = chi_squared(normalized_histogram_1, normalized_histogram_2)
  numerator = (normalized_histogram_1 - normalized_histogram_2) .^ 2;
  denominator = normalized_histogram_1 + normalized_histogram_2;
  distance = 0.5 * sum(sum(numerator ./ denominator));
end
```

```
% Author: <njoffe@ucsd.edu> Nitay Joffe
% Date: 10/24/2006
% Class: CSE 166 - Image Processing
% Homework: 1
% Problem: 3 - Color Histograms
clear;

% (b) Using the above function, compute and display the color histogram for each
%     of five different rectangular regions of your choice for Figure 6.30(a)
%     (a bowl of strawberries). Use bar to plot each histogram, and clearly mark
%     the x-axis to indicate the division between the R, G, and B intervals.
%     Annotate your plot (use gtext or just mark it by hand) to explain the
%     significance of the various peaks you observe. Along with each histogram,
%     display the corresponding region of the image used to compute it.
number_of_regions = 5;

rgb_image = imread('Fig6.30(a).jpg');

figure;
imshow(rgb_image);
region_images = {};
region_histograms = {};
region_positions = {};
for i = 1:number_of_regions
  [region_images{i},region_histograms{i},region_positions{i}] = rgb_histogram(rgb_image);
end

figure;
for i = 1:number_of_regions
  subplot(number_of_regions,2,i*2-1);
  imshow(region_images{i});
  subplot(number_of_regions,2,i*2);
  bar(region_positions{i},region_histograms{i});
end

% (c) Use this function to compare all pairs of the five histograms you computed
%     in the previous step. Output the values in a 5 by 5 matrix; because of
%     symmetry, only the upper triangular portion of this matrix is needed.
%     Choose at least three representative entries in this matrix and explain
%     their values in terms of the color distributions you compared to obtain
%     them.
chi_squared_distance = [];
for i=1:number_of_regions
  for j=1:number_of_regions
    chi_squared_distance(i,j) = chi_squared(region_histograms{i},region_histograms{j});
  end
end
chi_squared_distance
```