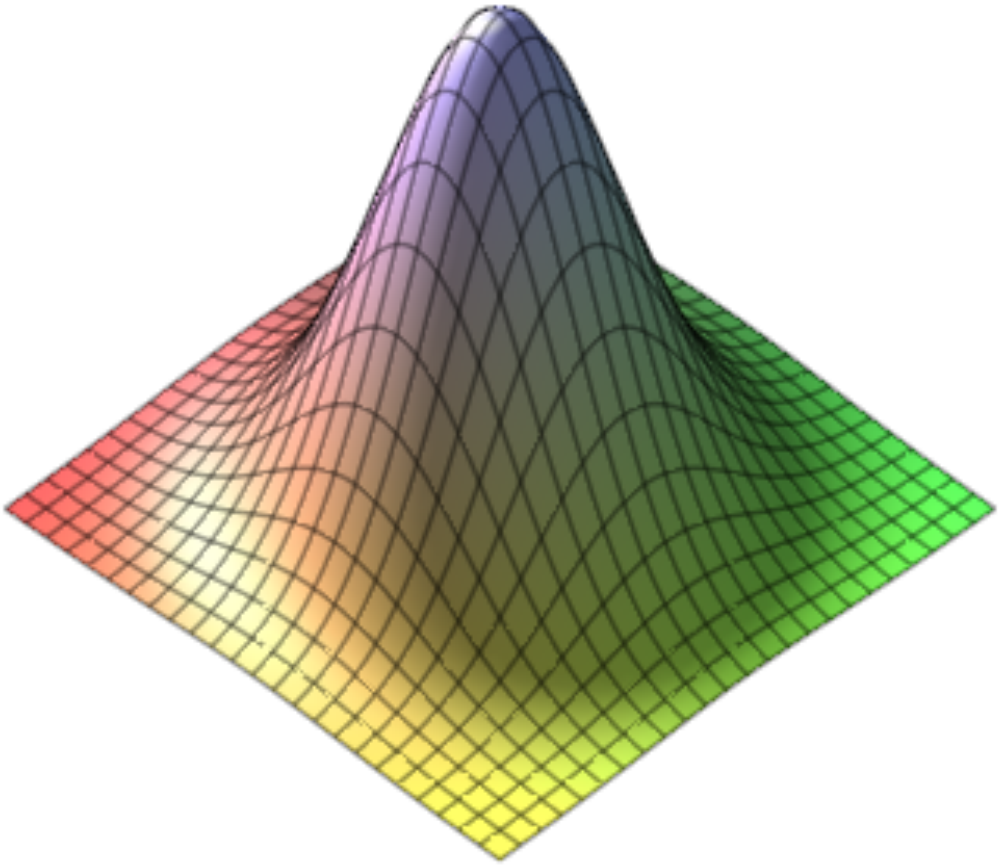

Visual Computing 2015

Image Processing Assignment

Due: 19 October

GETTING STARTED

Input Images

Download images from the course website to create your own image gallery. You will be using these images for generating the results of your assignment. For each part of the assignment you should show your results using all of these images.

Matlab

For this assignment you should use Matlab. UCI provided free Matlab for students. If you don't have Matlab you can download and install it using your own UCInetID. In the following link you can find the instruction of installing student version of Matlab for UCI:
http://laptops.eng.uci.edu/software-installation/matlab

Deliverables

Please use EEE dropbox to upload your files. You should upload a PDF file showing your results and a Zip file including all your codes. For writing the PDF document make sure to follow the instruction for each question and provide your results with the requested format in each part of the assignment.

ASSIGNMENT

Part 0: Convolution

Try to write a code to convolve a Gaussian Filter on given input images. As a result you should get a blurred image. For this purpose you can use *imfilter* function in Matlab and you don't need to implement the convolution. Try to change the parameters of the Gaussian Filter and compare the results.
For this part of the assignment you don't need to submit anything. But you need to know how to use convolution in Matlab to do next parts of the assignment.

Part 1: Gaussian Pyramid

Write a program to generate Gaussian pyramid using convolution. Generate each level of the pyramid by applying a 2x2 box filter to the image in the immediately preceding level.

1/4	1/4
1/4	1/4

Pick your test pictures with size of 2^N in each dimensions. Then use above filter in each level and produce a smaller image with half the size of the image in previous level in each dimension. At the end you should get a 1 by 1 image with the color that is the average of all pixels in original image.
Because it is hard to illustrate the smaller images in the next step you need to create images with the same size at each level.
For this purpose use the same filter as Step 1 but resample each level using bilinear interpolation to produce an image with the same size as the original image. So you can see images with different levels of blurring in each level and at then end you should get an image with the same size as original image for each level of Gaussian Pyramid.
Create the Gaussian pyramid for all the images in the gallery and put the images in the PDF file. Please consider that all the images in different levels should have same size.

Part 2: Laplacian Pyramid

Write a program to generate the Laplacian pyramid by subtracting the consecutive levels of the Gaussian pyramid. Create the Laplacian pyramid for all the images in the gallery and put the results in the PDF document.

Part 3: Multi-Scale Edge Detection

Step 1: Generate the *second order derivative images* at different scales (or resolution) using a Laplacian operator given below:

-1/8	-1/8	-1/8
-1/8	1	-1/8
-1/8	-1/8	-1/8

Apply the Laplacian operator to every level of the Gaussian pyramid generated in the part 1.
Step2: Segment the second order derivative image by assigning value 1 to all pixels of magnitude greater than 0 and value 0 to all pixels of magnitude less than or equal to zero.
Step 3: Detect the zero crossing in the segmented image. This is done by tagging any pixel which has at least one neighbor who is of different value than the pixel itself.
Step 4: Examine the pixels surrounding the zero crossing pixels in the second order derivative image. Calculate the local variance and mark it as an edge pixel if this value is greater than a certain threshold. This completes the edge detection.
Show the result for all the images in the gallery in your PDF document.

Part 4: Multi-Resolution Spline

Use your code for creating Gaussian and Laplacian pyramid to blend two image using following steps:
Step 1: Create the Laplacian pyramid for both images.
Step 2: Create a mask to use for blending images.
Step 3: Create the Gaussian pyramid for the above mask.
Step 4: Now create a new Laplacian pyramid by linear interpolation of two Laplacian pyramids from step 1. For combining each pixels of images in each level of Laplacian pyramids use the value of the generated Gaussian pyramid from step 3 as the coefficient of linear interpolation.
Step 5: Now add the images of all levels of the pyramid from step 4.

Choose 3 pairs of images from the gallery. and show the original images and your mask and the final result in you PDF file.