Computational Photography HW3

contents:

1. introduction

In this homework assignment, we will be putting together a pyramid blending pipeline that will allow us to turn the following three images,

Into this hideous creature. (Hopefully you will be able to create some less disturbing images using your own photographs).

For details on these images, see the images/IMAGES.txt file in the assignment folder.

These are the sample images which are provided for you in the images/source/sample subdirectory. The code in **run.py** will scan through all the folders in the images/source directory. Within each folder, it will look for images with filenames that contain ‘white’, ‘black’ and ‘mask’. Once it finds a folder with all three such images, it will apply the blending procedure to them, and save the output to images/output/. Along with the output image, it will create visualizations of the gaussian and laplacian pyramids used in the process.

The blending procedure takes the two images and the mask, and splits them into their red, green and blue channels. It then blends each channel separately.

As discussed in lecture, the code will construct a laplacian pyramid for the two images. It will then scale the mask image to the range [0,1] and construct a gaussian pyramid for it. Finally, it will blend the two pyramids and collapse them down to the output image.

Pixel values of 255 in the mask image are scaled to the value 1 in the mask pyramid, and assign the strongest weight to the image labeled ‘white’ during blending. Pixel values of 0 in the mask image are scaled to the value 0 in the mask pyramid, and assign the strongest weight to the image labeled ‘black’ during blending.

In order to facilitate this process, you will be providing six (6) key functions through this programming assignment. First, you will implement the reduce and expand functions in **part0.py**. Then, you will write code to construct gaussian and laplacian pyramids using these functions in **part1.py**. Finally, you will write code that blends two laplacian pyramids using a mask, and collapses a laplacian pyramid into an output image in **part2.py**.

We expect that you already have python and required packages installed. If not, consult the ‘software installation’ link on the left hand side of the coursera page. We expect this assignment to take you less than 5 hours.

2. part 0

As with previous assignments, running part0.py directly will apply a unit test to your code and print out helpful feedback. You can use this to debug your functions. Once you are done, you can submit your functions using submit.py.

reduce

This function is defined in the part0.py file. The lecture and tutorial videos, and the documentation string in the file will give you more detail about what it should do.

As seen in the lectures and tutorial video, this function takes an image and subsamples it down to a quarter of the size (dividing the height and width by two). Before we subsample the image, however, we need to first smooth out the image.

Within the code, you are provided with a generating_kernel(a) function. This function takes a floating point number **a**, and returns a 5x5 generating kernel. For the reduce and expand functions, you should use a=0.4.

Seeing as you already know how the convolve function works, for this assignment you should use the library implementation of convolve,

```
import scipy.signal
scipy.signal.convolve(image, kernel, 'same')
```

This call will convolve the image and kernel, and return an array of the 'same' size as image. See the references section for more details on this function.

expand

This function is defined in the part0.py file. The lecture and tutorial videos, and the documentation string in the file will give you more detail about what it should do.

As seen in the lectures and tutorial video, this function takes an image and supersamples it to four times the size (multiplying the height and width by two). After increasing the size, we have to interpolate the missing values by running over it with a smoothing filter.

For this part of the assignment, please use the generating kernel with a=0.4, and the convolve2d function from the reduce function discussion above.

3. part 1

In this part of the assignment, you will be implementing functions that create gaussian and laplacian pyramids. As usual, use part1.py to test your code. In addition, run.py defines the functions viz_gauss_pyramid and viz_lapl_pyramid, which take a pyramid as input and return an image visualization. You might want to use these functions to visualize your pyramids while debugging.

gauss_pyramid

This function is defined in the part1.py file. The lecture and tutorial videos, and the documentation string in the file will give you more detail about what it should do.

This function takes an image and builds a pyramid out of it. The first layer of this pyramid is the original image, and each subsequent layer of the pyramid is the reduced form of the previous layer.

Within the code, these pyramids are represented as lists of arrays, so

```
pyramid = [layer0, layer1, layer2, ...]
layer0 = np.array(shape, dtype = float)
```

Please use the reduce function that you implemented in the previous part in order to write this function.

lapl_pyramid

This function is defined in the part1.py file. The lecture and tutorial videos, and the documentation string in the file will give you more detail about what it should do.

This function takes a gaussian pyramid constructed by the previous function, and turns it into a laplacian pyramid. The doc string contains further information about the operations you should perform for each layer.

Like with gaussian pyramids, laplacian pyramids are represented as lists of numpy arrays in the code.

Please use the expand function that you implemented in the previous part of the code in order to write this function.

4. part 2

In this part, you will be completing the pipeline by writing the actual blend function, and creating a collapse function that will allow us to disassemble our laplacian pyramid into an output image.

As always, you can use part2.py to test your code. Once you are finished, submit the code with submit.py.

blend

This function is defined in the part2.py file. The lecture and documentation string in the file will give you more detail about what it should do.

This function takes three pyramids:

white - a laplacian pyramid of an image
black - a laplacian pyramid of another image
mask - a gaussian pyramid of a mask image

It should perform an alpha-blend of the two laplacian pyramids according to the mask pyramid. So, you will be blending each pair of layers together using the mask of that layer as the weight.

As described in the doc string, pixels where the mask is equal to 1 should be taken from the white image, pixels where the mask is 0 should be taken from the black image. Pixels with value 0.5 in the mask should be an equal blend of the white and black images, etc.

You may assume that all of the provided pyramids are of the same dimensions, and have dtype float. You may further assume that the mask pyramid has values in the range [0,1]

Your output pyramid should be of the same dimension as all the inputs, and dtype float.

collapse

This function is defined in the part2.py file. The documentation string will give you more detail about what it should do.

This function is given a laplacian pyramid, and is expected to ‘flatten’ it to an image.

We need to take the top layer, expand it, and then add it to the next layer. This results in a pyramid of one level less than we started with. We continue this process until we are left with

only a single layer.

This function should return a numpy array of the same shape as the base layer of the pyramid, and dtype float.

5. references

numpy and scipy - link
numpy user guide - link
opencv - link
scipy.signal.convolve2d - link