CPE 428: Computer Vision
Homework 3: Scale-Invariant Feature Detection
Instructor: Jonathan Ventura

In this assignment you will implement a script to detect scale-invariant features, also known as difference-of-Gaussian or DOG features. Such features are the foundation for many highly-successful computer vision systems for tasks such as object recognition and 3D reconstruction. The famous Scale-Invariant Feature Transform (SIFT) algorithm by David Lowe uses a DOG detector similar to what you will implement.

First, you will build a DOG response stack by applying successively larger Gaussian filters to the input image, and taking the difference between neighboring filter responses. The difference images constitute the DOG responses (the "layers" of the stack). Features are detected as local minima and maxima in the DOG response stack, subject to a threshold on the absolute value of the DOG response.

Note: you can use `skimage.ndimage.gaussian_filter` to perform the Gaussian filtering. This is a bit easier than using OpenCV functions to do it.

You can implement non-maxima suppression by hand, but it will probably be extremely slow. The code template describes a fast way to compute it using `skimage.ndimage.minimum_filter` and `skimage.ndimage.maximum_filter`.

Note that in Numpy you can create a Boolean array by doing operations such as `(response == 5)`. This will create a Boolean array the same size as `response`, that is equal to `True` wherever the value in the `response` array equals 5. You can also use Boolean logic on these arrays, such as : `(response > 0) & (response < 5)`. Finally, `np.argwhere(a)` will return a list of all the locations where `a` is true.

Implement all the functions in detector.py, and test out your system on the provided example images.

Once you have the feature detector implemented and working, inspect the results and answer the following qualitative evaluation questions:

1. What is the effect of changing the `thresh` parameter? Show and analyze example results where `thresh` is increased and decreased from the default.
2. What is the effect of increasing the `sigma` parameter? Show and analyze example results where `sigma` is increased from the default.
3. Using an image of your choice (from the internet or captured by you on your phone), first hypothesize where you think features will be found, and then compare to what the detector actually finds.

Required libraries:

- cv2 (OpenCV)
- numpy
- sckit-image

Links to example inputs and outputs:

- sunflowers
- sunflowers result
- rocks
- rocks result

Deliverables:

- process.py
- detector.py
- Text document (plain text, Word doc, or PDF) explaining your solution

Please upload these files individually, not as a ZIP – this makes grading easier.

Deadline: Sunday, Nov 12, 11:59pm