Introduction

Image deconvolution is the opposite process to convolution, or the removal of an image filter. Many of our scientific image observations come in through some convolved image process, usually from some physical properties associated with light diffraction. Common examples of this are found in astronomical observations where light is warped around large bodies of mass and gravity causes light to scatter in various ways. Another common example is with light microscopy. This suffers from the other physical effects of quantum mechanics and wave behavior of light at miniscule scales.

The point of image deconvolution is to find out what the actual ground truth part of the image is. Much of the algorithms that are used today rely on the properties of convolutions in the fourier space. The deconvolution operation can be represented by the division of the obtained image by the point spread function (PSF is another word for the filter that is moved over the image). Two things complicate this process however, noise in the images that can't be attributed to the PSF, and the measurement of a PSF.

Blind deconvolution is the process by which you determine the ground truth image without knowing exactly what the PSF is to begin with. This requires the use of bayesian statistics to maximize the likelihood that the generated PSF was responsible for creating the convolved image, while simultaneously deconvolving the image.

Current Implementations

Currently in python the well known package skimage has an implementation of regular deconvolution which can be found here. Skimage uses the function signature

richardson_lucy(*image*, *psf*, *num_iter=50*, *clip=True*, *filter_epsilon=None***)**

There aren't many blind deconvolution implementations in python, and those that do have a ton of other things wrapped into it, like the imageJ wrappers. The only thing that I could find was an implementation of blind deconvolution in MATLAB. Luckily there is also a MATLAB wrapper available in python that allows you to run MATLAB code natively.

MATLAB Wrapper

The MATLAB wrapper works by using the MATLAB array types which are transformed into C types. These are then fed into the MATLAB Python engine which is run and sent back through the same pipeline back to a MATLAB array type response. Luckily numpy can handle these MATLAB arrays and convert them to a more usable format.

Using the wrapper, I will call the MATLAB blinddeconv function. This seemed to be the best way to do it given that the MATLAB function has a bunch of mathematical niceties built into it which will make the image quality better than what I could have from scratch. This also goes along with the whole theme of the final project better in terms of making something easier to use.

Implementation

At the top of the module I have the pylab package which holds a subpackage for images. A submodule inside of images is made which contains the restoration submodule and this is where the blind deconvolution functions will lie. Since there is no need for initialization, the script is only to contain the function for deconvolution and a global variable for the matlab

engine object. The function can take various image forms, such as a numpy array of an image, a PIL Image class, or even a string that contains a URL to a target image. I think that this makes the function a lot more versatile than the skimage version while still keeping it simple. I also kept the output the same as the skimage implementation instead of adding the predicted PSF.

Software Requirements

For this project you need a working installation of MATLAB 2020b or later as well as the Image Processing Toolbox package. In addition you need either Python 3.6, 3.7,or 3.8. For Python libraries I have included a requirements.txt file that has all of the necessary packages and their versions.

Use Cases

Could be especially useful for very large datasets that can't be held in memory all at once, so you have to process them individually. Deconvolution is usually one of the first steps in an image pipeline since it refines image quality and improves the effectiveness of downstream tasks. This package would provide an easy way to iterate over images through say a PyTorch data loader before feeding them into a deep learning model.

I also kept the use of the package in smaller settings like single images in say a Jupyter notebook. You can have the image in memory and pass it into the function as well as use the show function to display the image in the notebook. Another sub function is the save function which allows you to pass in a URL where you would like to save the deconvolved image.

Extensibility

This software is very easily extendable because of the MATLAB wrapper that is built around it. MATLAB has a ton of scientific functions that are well tested and that might not occur in any large python libraries. The package setup that I have created is also good for adding on different modules that may have functionalities outside the image restoration domain. This is just one small use case of implementing MATLAB's functionality but there is tons of stuff in things such as dynamic systems and simulation that I am sure could be ported over. I have it setup so it's the main package at the top, followed by the functional domain, followed by the operations themselves. So adding a fluid simulation would follow something like pylab.simuation.fluid.finite_volume.function_name().