MIT EECS 6.815/6.865: Assignment 9:

# Optical Correction

Due Sunday April 15 at 9pm

## 1 Summary

- Chromatic aberrations

## 2 Lateral Chromatic Aberrations

We will correct for lateral chromatic aberration by scaling each channel separately with respect to the center of the image. Using a black-and-white calibration pattern, we will compute the scale factor for the red and blue channel so that they match the green channel.

### 2.1 Scaling using scipy

Write a function `scaleChannel(im, i, k)` that returns a new image based on `im` but where channel `i` has been scaled by a factor of `k` with respect to the image center (i.e. leaving the center pixel unchanged.)

For speed and quality, we require that you use scipy's `ndimage.map_coordinates(channel, coord`, which takes as input a a 2D array `channel` and a an array `coord` storing the source coordinates where the image should be resampled. `coord` is a 3D array where the first dimension has size two (the first element stores y coordinates and the second one the x coordinates) and the remaining dimensions correspond to the source y or x values for that particular output pixel location. That is, output pixel at $y_{out}, x_{out}$ will be resampled at the source location $y_{src} =$`coord`$[0, y_{out}, x_{out}]$, $x_{src} =$`coord`$[1, y_{out}, x_{out}]$. One advantage of `ndimage.map_coordinates` is that it performs bicubic reconstruction by default.

`numpy` has a function `mgrid[0:h, 0:w]` that can be handy to initialize these coordinates for the identity resampling. You can then use array operations to appropriately scale the coordinates. Don't forget that the center is the center of scale, which does not correspond to $0, 0$.

Try your function with large-enough scale factors to debug it.

### 2.2 Error function

Write a function `lateralChromaError(im, i)` that computes the misalignment between channel `i` and channel 1 (green). First, normalize channel `i` to have the same average as channel 1 (grey-world white balance). Then compute the

sum of the square difference between the resulting channel and 1. This is your error.

You should be able to use only array operations for this.

### 2.3 Calibration

Write a function `findChroma(im, i, maxPixel=3, steps=20)` that searches for the best scale factor to remove chromatic aberrations of channel `i` by scaling it until it aligns with channel 1. Use a brute force search. The largest and smallest scale factor you consider should correspond to a shift of `maxPixel` pixels at the periphery (that is, k goes between 1+/-maxPixel/max(height, width)). Try `steps` different values. Just in case, note that `numpy` has a function `arange` that performs like `range` but for floating point values.

### 2.4 Putting it all together

Use the included image `24.png` to calibrate Canon's 24-105 f/4 IS L at its widest angle setting. Compute the scale factor both for the red and the blue channel. Use it to enhance the included images.

### 2.5 6.865: Calibrate your own lens

Take a picture of the checkerboard calibration pattern and calibrate a different lens. Report your scale factors and apply the correction to an image taken with the same lens at the same focal length.

## 3 Extra credits

Binary search (easy)

Quasi-Newton based on finite difference.

Quasi Newton based on image gradient (Lucas Kanade)

Radial distortion

Vignetting

## 4 Submission

Turn in your images, python files, and make sure all your functions can be called from a module `a8.py`. Put everything into a zip file and upload to Stellar.

Include a `README.txt` containing the answer to the following questions:

- How long did the assignment take?
- Potential issues with your solution and explanation of partial completion (for partial credit)
- Any extra credit you may have implemented

- Collaboration acknowledgement (but again, you must write your own code)
- What was most unclear/difficult?
- What was most exciting?
- scale factors for the included image
- information about your lens and scale factors.

Images:

- Results for the two included images
- 6.865: results for your own image and for the calibration image.