CS543 Assignment 2

**Your Name:** <name>

**Your NetId:** <netid>

# Part 1 Fourier-based Alignment:

You will provide the following for each of the six low-resolution and three high-resolution images:

* Final aligned output image
* Displacements for color channels
* Inverse Fourier transform output visualization for ***both*** channel alignments ***without*** preprocessing
* Inverse Fourier transform output visualization for ***both*** channel alignments ***with*** any sharpening or filter-based preprocessing you applied to color channels

You will provide the following as further discussion overall:

* Discussion of any preprocessing you used on the color channels to improve alignment and how it changed the outputs
* Measurement of Fourier-based alignment runtime for high-resolution images (you can use the python time module again). How does the runtime of the Fourier-based alignment compare to the basic and multiscale alignment you used in Assignment 1?

## A: Channel Offsets

Replace <C1>, <C2>, <C3> appropriately with B, G, R depending on which you use as the base channel. Provide offsets in the **original image coordinates** and be sure to account for any cropping or resizing you performed.

Low-resolution images (using channel <C1> as base channel):

|  |  |  |
| --- | --- | --- |
| Image | <green> (h,w) offset | <red> (h,w) offset |
| 00125v.jpg | (2, 5) | (1, 10) |
| 00149v.jpg | (2, 4) | (2, 9) |
| 00153v.jpg | (2, 7) | (4, 14) |
| 00351v.jpg | (0, 4) | (0, 13) |
| 00398v.jpg | (3, 5) | (4, 11) |
| 01112v.jpg | (0, 0) | (1, 5) |

High-resolution images (using channel <C1> as base channel):

|  |  |  |
| --- | --- | --- |
| Image | <green> (h,w) offset | <red> (h,w) offset |
| 01047u.tif | (19, 24) | (32, 70) |
| 01657u.tif | (9, 56) | (13, 120) |
| 01861a.tif | (39, 70) | (63, 146) |

## B: Output Visualizations

For each image, insert 5 outputs total (aligned image + 4 inverse Fourier transform visualizations) as described above. When you insert these outputs be sure to clearly label the inverse Fourier transform visualizations (e.g. “G to B alignment without preprocessing”).

### 00125v.jpg

A picture containing grass, water, sky, outdoor

Description automatically generated

Graphical user interface

Description automatically generated with low confidence

A picture containing diagram

Description automatically generated

### 00149v.jpg

A painting in a frame

Description automatically generated with medium confidence

A picture containing shape

Description automatically generated

A picture containing shape

Description automatically generated

### 00153v.jpg

A picture containing person

Description automatically generated

A picture containing graphical user interface

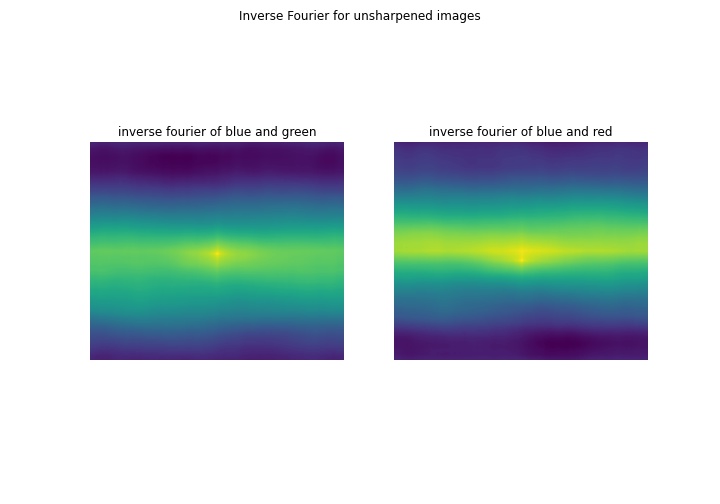
Description automatically generated

A picture containing graphical user interface

Description automatically generated

### 00351v.jpg

### A picture containing grass, outdoor, old, house Description automatically generated



A picture containing shape

Description automatically generated

### 00398v.jpg

A train on the railway tracks

Description automatically generated

Shape

Description automatically generated

Shape

Description automatically generated with medium confidence

### 01112v.jpg

A white building with a domed roof

Description automatically generated with low confidence

Graphical user interface

Description automatically generated

A picture containing shape

Description automatically generated

### 01047u.tif

A group of vases sit on a shelf

Description automatically generated with low confidence

A picture containing graphical user interface

Description automatically generated

A picture containing graphical user interface

Description automatically generated

### 01657u.tif

A person sitting on a couch

Description automatically generated with medium confidence

Diagram

Description automatically generated with low confidence

Diagram

Description automatically generated with low confidence

### 01861a.tif



A picture containing square

Description automatically generatedA picture containing square

Description automatically generated

## C: Discussion and Runtime Comparison

For color channels preprocessing, I used crop function to crop the black boarder around the 3 color channel images. And before processing FFT, I used gaussian filter to sharpen the 3 color channels images. Both 2 preprocessing improved the accuracy of image alignment.

The running time of Fourier-based alignment is around 7 seconds for each high-resolution image, which is around 3.9 times faster than SSD multiscale alignment, 9.4 times faster than basic SSD alignment. The running time of SSD multiscale solution for each high-resolution image is around 27 seconds, which is around 2.4 times faster than basic SSD alignment solution. When using basic SSD alignment method, it will take around 66 second to colorize each image.

# Part 2 Scale-Space Blob Detection:

You will provide the following for ***8 different examples***(4 provided, 4 of your own):

● original image

● output of your circle detector on the image

● running time for the "efficient" implementation on this image

● running time for the "inefficient" implementation on this image

You will provide the following as further discussion overall:

● Explanation of any "interesting" implementation choices that you made.

● Discussion of optimal parameter values or ones you have tried

### Example 1:

Original Image



Output of increase filter size method: Runtime: 25.10 seconds



Output of down sampling method: Runtime: 0.803 seconds

A picture containing text

Description automatically generated

### 

### Example 2:

Original Image

A person writing on a chalkboard

Description automatically generated

Output of increase filter size method: Runtime: 30.77 seconds

A picture containing text

Description automatically generated

Output of down sampling method: Runtime: 1.212 seconds

A picture containing text

Description automatically generated

### 

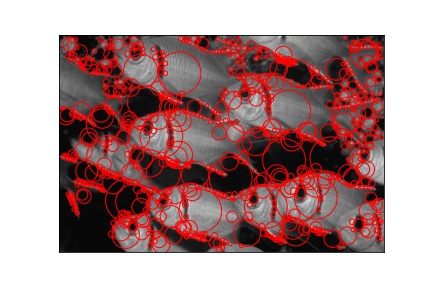
### Example 3:

Original Image

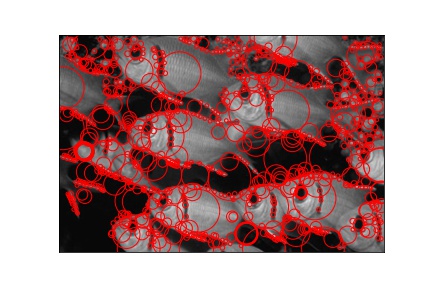
A picture containing fish, spiny-finned fish

Description automatically generated

Output of increase filter size method: Runtime: 21.61 seconds



Output of down sampling method: Runtime: 0.712 seconds



### Example 4:

Original Image

A field of yellow flowers

Description automatically generated with medium confidence

Output of increase filter size method: Runtime: 19.73 seconds

A picture containing text

Description automatically generated

Output of down sampling method: Runtime: 0.499 seconds

A picture containing text

Description automatically generated

### Example 5:

Original Image

A picture containing indoor, arranged

Description automatically generated

Output of increase filter size method: Runtime: 13.91 seconds

A picture containing text

Description automatically generated

Output of down sampling method: Runtime: 0.275 seconds

A picture containing text

Description automatically generated

### 

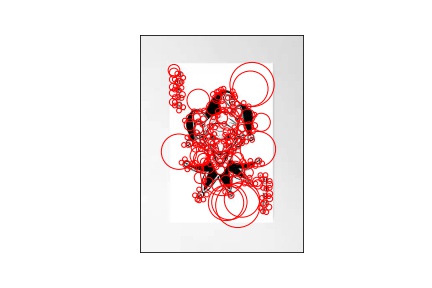
### Example 6:

Original Image

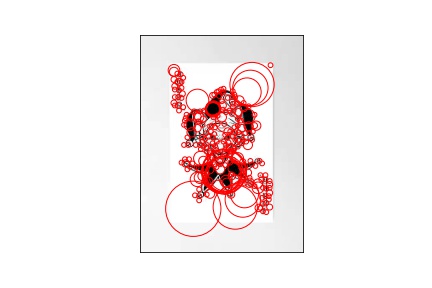
Whiteboard

Description automatically generated

Output of increase filter size method: Runtime: 13.93 seconds



Output of down sampling method: Runtime: 0.273 seconds



### Example 7:

Original Image

A group of bees on a honeycomb

Description automatically generated with medium confidence

Output of increase filter size method: Runtime: 14.06 seconds

A picture containing background pattern

Description automatically generated

Output of down sampling method: Runtime: 0.379 seconds

A picture containing background pattern

Description automatically generated

### Example 8:

Original Image

Background pattern

Description automatically generated

Output of increase filter size method: Runtime: 13.91 seconds

Chart, scatter chart

Description automatically generated

Output of down sampling method: Runtime: 0.297 seconds

Chart, scatter chart

Description automatically generated

## Discussion:

The speed of down sampling method for blob detection is about 0.7 second for each image, and the speed of increase filter size is about 30 second for each image. And the optimal thresholds for my down sampling method is 0.00015, and the optimal thresholds for my increase filter size method is 0.0028. The optimal initial sigma is 2, the optimal levels is 12, the optimal k is 1.25.

# Bonus:

## Blob-Detection Extra Credit

● Discussion and results of any extensions or bonus features you have implemented for Blob-Detection