GNR 602 Programming Assignment

Implement a PCA based pan-sharpening algorithm.

Vivek Veer (200050158),
Poorna Teja (200050101),
Jithendra (200050068)

What is pan-sharpening

- Pan-sharpening is a technique used in remote sensing, satellite imagery to create a high-resolution color image by combining a high-resolution panchromatic (black and white) image with a lower resolution multispectral (colored) image.
- Pan-sharpening works by fusing the spatial resolution of the panchromatic image with the spectral information of the multispectral image. This is done by taking advantage of the fact that the panchromatic image has a higher spatial resolution than the multispectral image, while the multispectral image has better spatial resolution.

PCA based approach to pan-sharpening

- PCA (principal component analysis) is a mathematical method that identifies the most important patterns (principal components) in a dataset. In the context of multispectral images, PCA can be used to extract the most important spectral information from the image.
- To perform PCA-based pan-sharpening, the first principal component of the multispectral image is extracted and replaced with the panchromatic image. Inverse PCA transform is applied to get the pan-sharpened image.
- By replacing the first principal component with the panchromatic image, the new, pan-sharpened image retains the most important spectral information from the original multispectral image, while also having a higher spatial resolution.

Our Approach

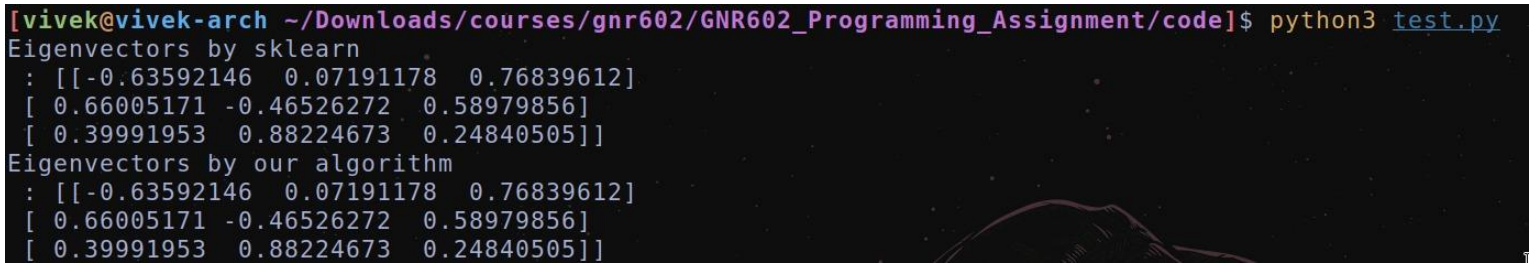
- Georeferencing is essentially the process of taking a digital image or dataset and giving it a specific location on the earth's surface. This is important because it allows us to overlay this information on top of other maps or satellite images and make meaningful comparisons or analyses.
- Our algorithm employs the correlation between the panchromatic image and the luminosity values in the multispectral image to enhance the spatial resolution of the multispectral image while preserving the color information, through a combination of PCA transformation and appropriate scaling of the luminosity values.

Our Approach

```
## Performing PCA on MS image (built our custom pca function)
pcs, eigs = util.pca_image(ms_img)
x = pcs
## Luminosity histogram matching for the panchromatic image as part of Contrast adjustment
pan_adjusted = (pan_img - pan_img.mean()) * (np.std(x) / pan_img.std()) + x.mean()
## Replacing the first component
x[:,:,0] = pan_adjusted
## Inverse PCA transform to get back an improved MS image
pan_sharpened = util.inverse_pca_image(x, eigs)
## Clipping the image to 0-255
pan_sharpened = np.clip(pan_sharpened, np.iinfo(dtype).min, np.iinfo(dtype).max)
## Converting floating pt. numbers to integer values in the pan sharpened image
pan_sharpened = pan_sharpened.astype(dtype)
```

Our Approach (verifying validity of our custom pca function)

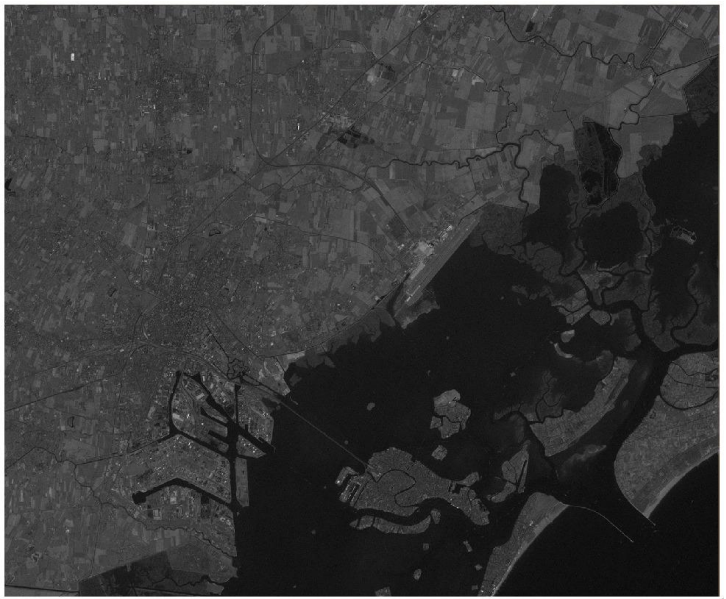
```
import numpy as np
import util
from sklearn.decomposition import PCA
X = np.random.rand(3, 3, 3)
## util.pca_image is our custom built PCA function
pcs, eigs = util.pca_image(X, 3)
X = np.reshape(X, (9, 3))
pca = PCA(n_components = 3)
pca.fit(X)
print(f"Eigenvectors by sklearn \n : {pca.components_.T}")
print(f"Eigenvectors by our algorithm \n : {eigs}")
```



Results

Our code and instructions on how to use it are available in this github repository.

Results



High resolution PC image

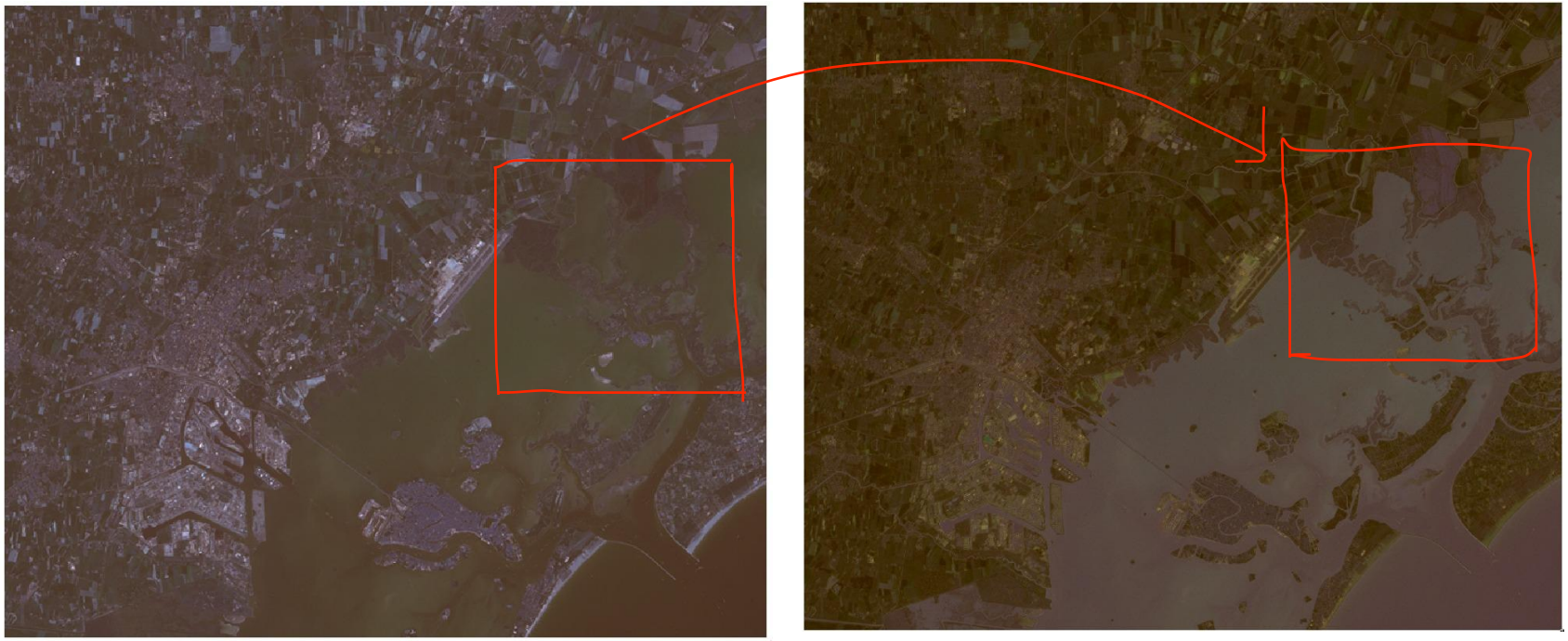
Low resolution MS image

Results

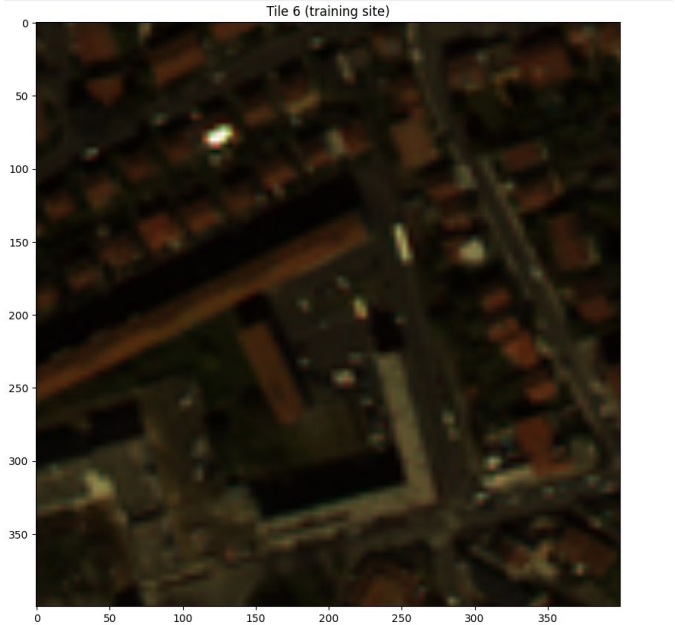
Pan sharpened MS image

Results

As an example notice the detail improvement
in the highlighted region



Results



Low resolution MS image

High resolution PC image

Results

As an example notice the detail improvement in the highlighted region

Low resolution MS image

Pan sharpened MS image