Lab 1: Data Processing with Python - BE 585

Danielle Tadych

Spring 2022

1/30/2022

Note: Skip the install cells if you've already installed the packages to the environment

--- Exercise 1 ---

In this exercise, you will:

a) Read tabular data from an Excel file

c) Inspect values by

- Pixel
- Row
- Column
- Window/Frame/Box

In [1]:
```
# Activating Environment and installing packages, will comment out later
# conda activate remsens

# Forgot that I already activated it lmao
```

In [2]:
```
#conda install numpy
```

In [3]:
```
#conda install pandas
```

In [4]:
```
#conda install openpyxl
```

In [5]:
```
# load library modules
import os
import xlrd
import numpy as np
import pandas as pd
import openpyxl
```

In [6]:
```
# Load the Excel file to be read using Pandas library
# also load the "RED" sheet for now inot a dataframe (Table)

df = pd.read_excel('Data/BE485_UofACampus.xlsx', sheet_name='RED')

# Define the Matrix size to read data into
nrows=6
ncols=10

# Create an Array data holder
DataRed=np.zeros((nrows,ncols))
```

In [8]:
```
# Now load data from the Excel Cells and store them into this 2D-array
# loop through each row and column to get the value
```

```
print("Extracting a small dataset from Excel...\n")

# For now we will use simple indexes to load the data form the frame
for i in range(0,nrows):
    for j in range(0,ncols):
        DataRed[i,j]=df.iloc[i,j]

# display all values read
print("The data array is:")
print(DataRed)

# get one single pixel/cell value and display it
Row = input("Enter the ROW number [0 based]: ")
Col = input("Enter the COLUMN number [0 based]: ")

pixelValue=DataRed[int(Row),int(Col)]
print("The pixel value at row=%3d and col=%3d is = " %(int(Row),int(Col)),pixel
```

```
Extracting a small dataset from Excel...

The data array is:
[[ 75.  42.  55.  81.  87.  75.  82.  82.  49.  13.]
 [ 75.  50.  44.  64.  75.  57.  55.  69.  68.  28.]
 [ 76.  67.  79.  97.  90.  83.  70.  62.  44.  30.]
 [ 45.  76.  84. 105. 107.  78.  77.  60.  44.  54.]
 [ 23.  57. 107. 112. 110.  91.  84.  61.  37.  62.]
 [ 17.  81. 113. 107. 117. 109. 101.  74.  44.  59.]]

The pixel value at row=  3 and col=  3 is =  105.0
```

In [9]:
```
# extract a full row
print("The values for row=1 are:")
#Pay attention to  how we addressed the full row (this is called slicing)
rowValues=DataRed[1,:]
print(rowValues)

# extract a full column
print("The values for column=6 are:")
#Pay attention to  how we addressed the full row (this is called slicing)
colValues=DataRed[:,6]
print(colValues)
```

```
The values for row=1 are:
[75. 50. 44. 64. 75. 57. 55. 69. 68. 28.]
The values for column=6 are:
[ 82.  55.  70.  77.  84. 101.]
```

To be completed by student

Can you get a subset of rows and columns, like the example below

In [10]:
```
print(DataRed)
print()

ex_subset = DataRed[2:5,2:7]
```

```
print("Subsetting rows 3-5 and columns 3-7(highlighted above):")
print(ex_subset)
print()

#display a message to know the program ended
print("program ended.")
```

```
[[ 75.  42.  55.  81.  87.  75.  82.  82.  49.  13.]
 [ 75.  50.  44.  64.  75.  57.  55.  69.  68.  28.]
 [ 76.  67.  79.  97.  90.  83.  70.  62.  44.  30.]
 [ 45.  76.  84. 105. 107.  78.  77.  60.  44.  54.]
 [ 23.  57. 107. 112. 110.  91.  84.  61.  37.  62.]
 [ 17.  81. 113. 107. 117. 109. 101.  74.  44.  59.]]

Subsetting rows 3-5 and columns 3-7(highlighted above):
[[ 79.  97.  90.  83.  70.]
 [ 84. 105. 107.  78.  77.]
 [107. 112. 110.  91.  84.]]

program ended.
```

--- Exercise 2 ---

In this exercise:

a) read a data from an Excel file

b) display images

- single band
- rgb models

In [11]:
```
#conda install matplotlib
```

In [23]:
```
#pip install vip_hci
```

In [1]:
```
# load library modules
import os
import xlrd
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import viplab_lib as vip
```

In [2]:
```
# Select and load the Excel sheets by name

#Sheet_Red=pd.read_excel (r'.\Data\BE485_UofACampus.xlsx', sheet_name='RED')
#Sheet_Green=pd.read_excel (r'.\Data\BE485_UofACampus.xlsx', sheet_name='GREEN
#Sheet_Blue=pd.read_excel (r'.\Data\BE485_UofACampus.xlsx', sheet_name='BLUE')

Sheet_Red = pd.read_excel('Data/BE485_UofACampus.xlsx', sheet_name='RED')
Sheet_Green = pd.read_excel('Data/BE485_UofACampus.xlsx', sheet_name='GREEN')
Sheet_Blue = pd.read_excel('Data/BE485_UofACampus.xlsx', sheet_name='BLUE')

print(Sheet_Red, Sheet_Green, Sheet_Blue)
```

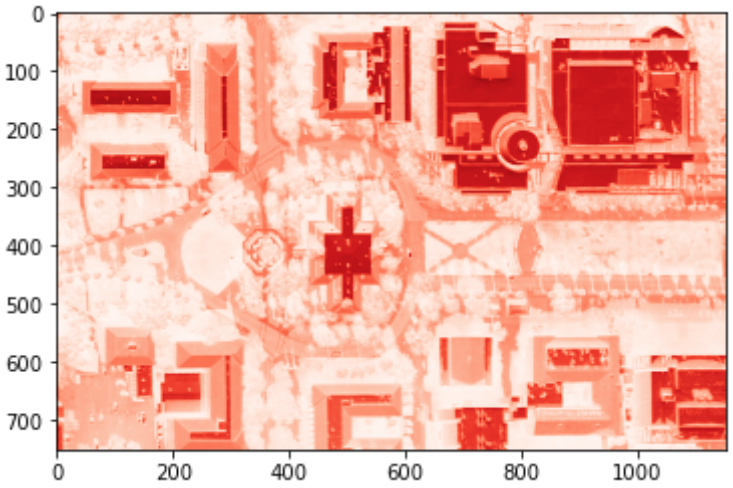
```
      71  71.1   64   84  84.1   81  104   67   32   19  ...  14.16  21.27  \
0     75    42   55   81    87   75   82   82   49   13  ...     26     36
1     75    50   44   64    75   57   55   69   68   28  ...     23     30
2     76    67   79   97    90   83   70   62   44   30  ...     24     26
3     45    76   84  105   107   78   77   60   44   54  ...     23     31
4     23    57  107  112   110   91   84   61   37   62  ...     17     20
..   ...   ...  ...  ...   ...  ...  ...  ...  ...  ...  ...    ...    ...
746  136   151  157  153   153  158  152  160  169  156  ...     94     90
747  118   144  160  157   152  151  144  154  168  151  ...     96     92
748  131   152  162  161   157  153  146  153  172  147  ...    103    101
749  156   162  157  160   160  156  150  153  167  152  ...     55     54
750  159   159  150  157   158  151  148  152  165  147  ...    128    130

     6.15  20.24  28.18  39.15  30.20  32.23  11.12  19.16
0      10     26     41     45     23     20     12     24
1      27     36     42     35     16     17     22     35
2      39     53     42     30     29     36     36     36
3      37     49     29      9      6     10      9      7
4      29     17     15     11      7      7      8      4
..    ...    ...    ...    ...    ...    ...    ...    ...
746    85     88     88     87     85     82     81     80
747    87     87     88     89     88     85     84     84
748    96     96     95     96     95     90     85     83
749    48     45     42     37     32     26     22     21
750   126    124    124    126    128    128    126    125

[751 rows x 1151 columns]       80  80.1   71   91   92   89  113   79   43
31  ...  21.19  31.22  \
0     84    51   62   88   94   83   91   93   60   25  ...     33     46
1     83    59   51   71   82   64   64   80   79   40  ...     33     40
2     84    75   85  104   97   90   77   70   55   42  ...     34     36
3     51    82   90  112  111   85   83   68   55   64  ...     35     41
4     29    63  113  118  114   95   90   67   45   72  ...     29     30
..   ...   ...  ...  ...  ...  ...  ...  ...  ...  ...  ...    ...    ...
746  131   148  156  156  158  165  158  163  171  153  ...     94     90
747  113   141  159  160  157  158  150  157  170  148  ...     94     90
748  126   149  161  164  162  160  152  156  174  144  ...    101     99
749  151   159  156  163  165  163  158  159  169  149  ...     52     50
750  154   156  149  160  163  158  156  158  167  144  ...    125    127

     16.11  27.18  35.19  43.14  34.18  32.19  11.13  17.14
0       20     33     48     50     27     20     12     22
1       37     43     49     40     21     17     22     33
2       49     60     49     35     34     36     36     34
3       47     56     36     14     11     10      9      5
4       39     24     22     16      9      7      8      2
..     ...    ...    ...    ...    ...    ...    ...    ...
746     85     88     88     87     85     82     81     80
747     85     87     88     89     88     85     84     84
748     94     94     95     96     95     90     85     83
749     46     43     40     35     30     23     19     18
750    123    122    122    124    126    125    123    122

[751 rows x 1151 columns]       63  63.1   53   73   71   68   94   59   27
17  ...  13.12  23.15  \
0     67    34   46   70   76   62   72   76   44   11  ...     25     38
1     68    42   35   53   64   46   47   63   63   26  ...     24     31
2     71    60   71   88   81   72   61   55   41   30  ...     25     27
3     41    70   76   96   96   69   69   53   41   53  ...     25     32
4     19    53  101  104  100   81   76   55   34   63  ...     19     21
```

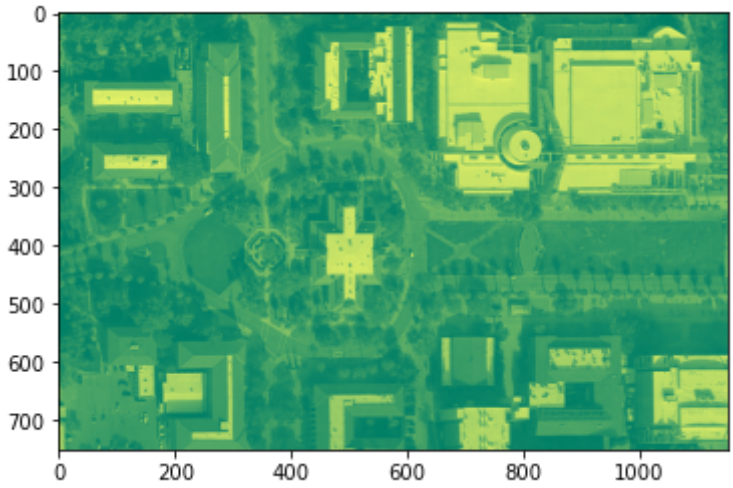
```
..   ...   ...  ...  ...  ...  ...  ...  ...  ...  ...  ...    ...    ...
746  125   141  151  149  151  157  148  154  160  144  ...     84     80
747  107   134  154  153  150  150  140  148  159  139  ...     82     78
748  120   142  156  157  155  152  142  147  163  135  ...     88     87
749  145   152  151  156  158  155  147  149  158  140  ...     37     38
750  148   149  144  153  156  150  145  148  156  135  ...    110    112

     8.14  20.20  28.21  42.7  33.15  34.20  13.13  22.24
0      12     26     41    46     26     22     14     27
1      28     36     42    36     17     17     22     36
2      40     53     42    31     30     36     36     37
3      38     49     29    10      7     10      9      8
4      30     17     15    12      6      7      8      5
..    ...    ...    ...   ...    ...    ...    ...    ...
746    75     76     76    75     73     70     69     68
747    73     75     76    77     76     75     74     74
748    82     82     83    84     83     80     75     73
749    33     30     27    23     18     14     10     11
750   108    109    109   112    114    116    114    115

[751 rows x 1151 columns]
```

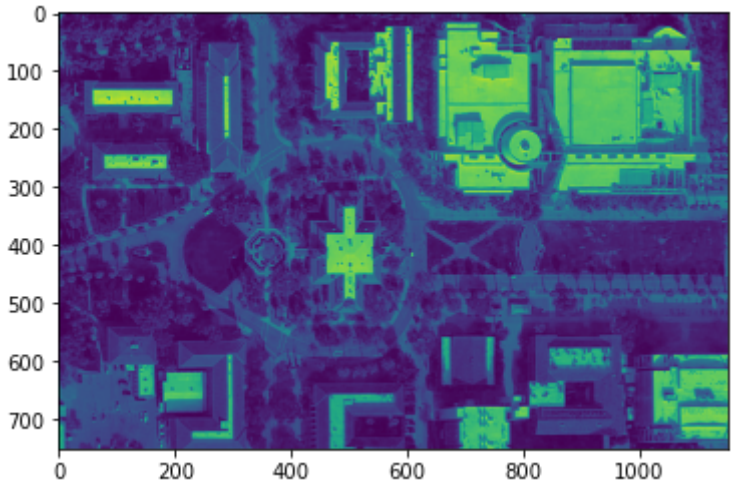
You can plot/show images with different colors using cmap = 'color table'
For a list of these colors maps refer to this page

In [3]:
```
# Display 'Red' band
plt.figure()
# plt.imshow(Sheet_Red,cmap='gray', vmin=0, vmax=255)
plt.imshow(Sheet_Red,cmap='Reds')

# Display 'Green' band
plt.figure()
#plt.imshow(Sheet_Green,cmap='viridis', vmin=0, vmax=255)
plt.imshow(Sheet_Green,cmap='summer')

# Display 'Blue' band
plt.figure()
#plt.imshow(Sheet_Blue,cmap='YlGn', vmin=0, vmax=200)
plt.imshow(Sheet_Blue,cmap='viridis')
```

Out[3]:
```
<matplotlib.image.AxesImage at 0x7ff77dbee760>
```







In [4]:
```
# Set the number of cells to read
nrows=500
ncols=500

# Create data holders for the data with the correct size
DataRed=np.zeros((nrows,ncols))
DataGreen=np.zeros((nrows,ncols))
DataBlue=np.zeros((nrows,ncols))
```

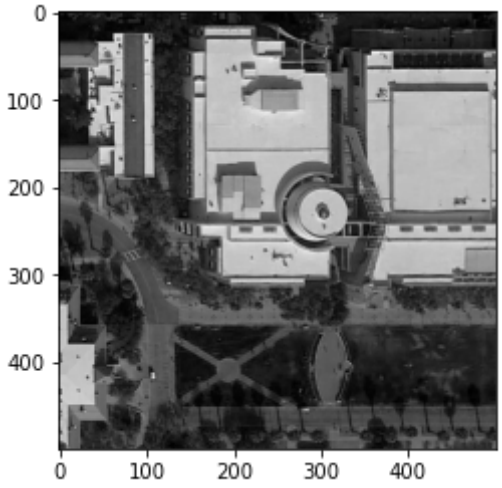
In [5]:
```
# Extract data from Excel Cells and store it in 2D-arrays
# loop through each row and column to get the value
print("Extracting data from Excel...")
for i in range(0,nrows):
    for j in range(0,ncols):
        DataRed[i,j]=Sheet_Red.iloc[i,j+500]
        DataGreen[i,j]=Sheet_Green.iloc[i,j+500]
        DataBlue[i,j]=Sheet_Blue.iloc[i,j+500]
print("complete")
```

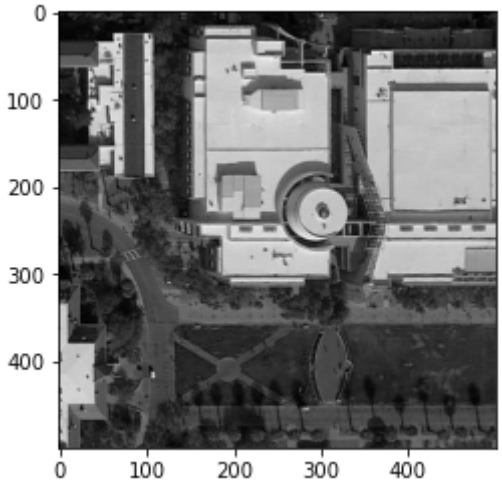
```
Extracting data from Excel...
complete
```

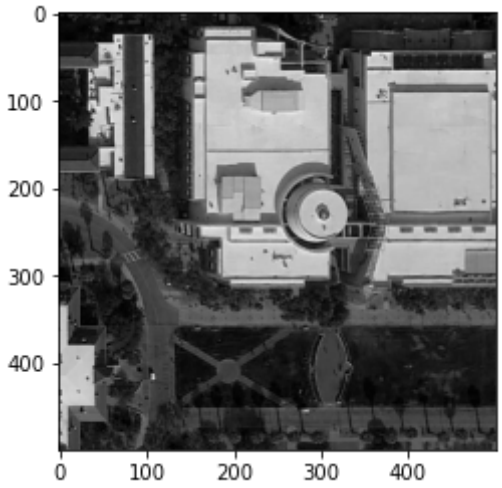
In [6]:
```
# Display 'Red' band
plt.figure()
plt.imshow(DataRed,cmap='gray')

# Display 'Green' band
```

```
plt.figure()
plt.imshow(DataGreen,cmap='gray')

# Display 'Blue' band
plt.figure()
plt.imshow(DataBlue,cmap='gray')
```

Out[6]: <matplotlib.figure.AxesImage at 0x7ff77de47f10>







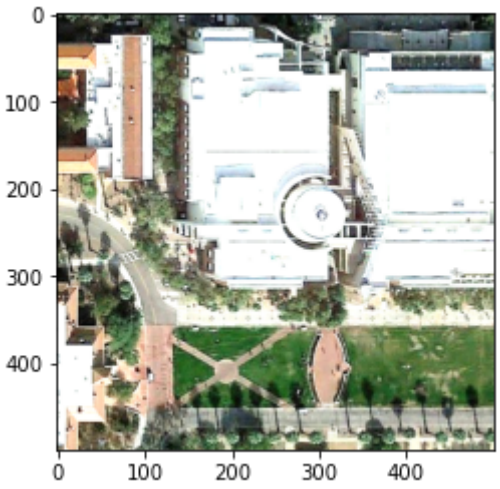
Here is another way using array Slicing

In [7]:
```
#DataRed=Sheet_Red.iloc[:500,500:1000]
```

```
#DataGreen=Sheet_Green.iloc[:500,500:1000]
#DataBlue=Sheet_Blue.iloc[:500, 500:1000]
```

In [8]:
```
# Combine all bands, the Red, Green and Blue data into an RGB model for display
print("Creating RGB Image...")
RGBImage=vip.Image_getRGB(DataRed,DataGreen,DataBlue,250)
# Display RGB Image
plt.figure()
plt.imshow(RGBImage)
print("done")
```

```
Creating RGB Image...
done
```



To be completed by student

- Now display the full RGB image, then
- Locate Old Main, Extract the corresponding data/window, and create an RGB image
- Place a color box around it (this will require some net-searching on your part if you do not know how that works

Full Image

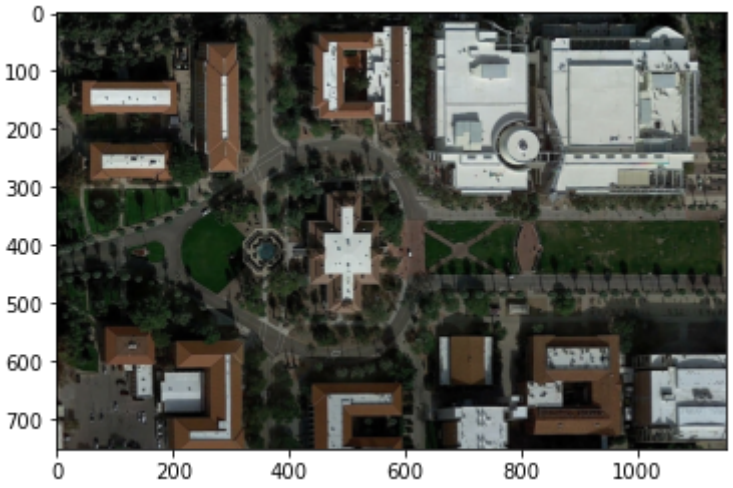
In [22]:
```
# [751 rows x 1151 columns]

nrows=751
ncols=1151

DataRed=np.zeros((nrows,ncols))
DataGreen=np.zeros((nrows,ncols))
DataBlue=np.zeros((nrows,ncols))

# Extract data from Excel Cells and store it in 2D-arrays
# loop through each row and column to get the value
print("Extracting data from Excel...")
for i in range(0,nrows):
    for j in range(0,ncols):
        DataRed[i,j]=Sheet_Red.iloc[i,j]
        DataGreen[i,j]=Sheet_Green.iloc[i,j]
        DataBlue[i,j]=Sheet_Blue.iloc[i,j]
```

```
print("complete")

print(type(DataRed),DataGreen,DataBlue)
```

```
Extracting data from Excel...
complete
<class 'numpy.ndarray'> [[ 84.  51.  62. ...  20.  12.  22.]
 [ 83.  59.  51. ...  17.  22.  33.]
 [ 84.  75.  85. ...  36.  36.  34.]
 ...
 [126. 149. 161. ...  90.  85.  83.]
 [151. 159. 156. ...  23.  19.  18.]
 [154. 156. 149. ... 125. 123. 122.]] [[ 67.  34.  46. ...  22.  14.  27.]
 [ 68.  42.  35. ...  17.  22.  36.]
 [ 71.  60.  71. ...  36.  36.  37.]
 ...
 [120. 142. 156. ...  80.  75.  73.]
 [145. 152. 151. ...  14.  10.  11.]
 [148. 149. 144. ... 116. 114. 115.]]
```

In [16]:
```
# Combine all bands, the Red, Green and Blue data into an RGB model for display
print("Creating RGB Image...")
Full_Image=vip.Image_getRGB(DataRed,DataGreen,DataBlue, 1200)
# Display RGB Image
plt.figure()
plt.imshow(Full_Image)
print("done")
```

```
Creating RGB Image...
done
```

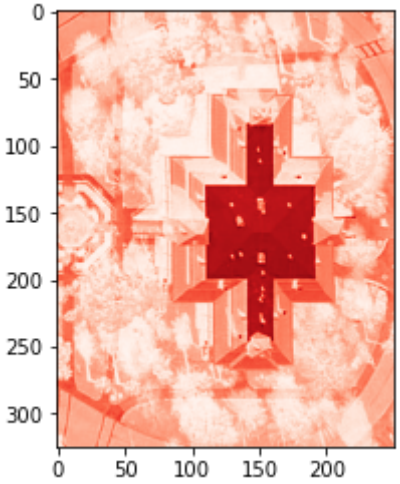


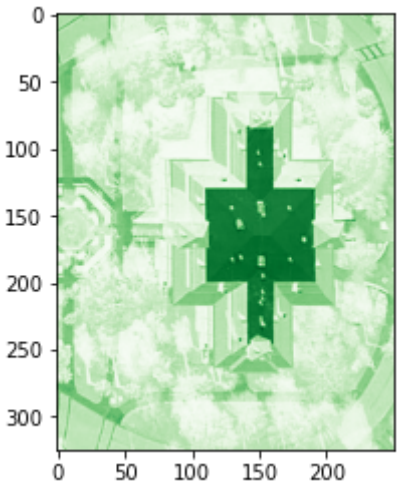
Old Main

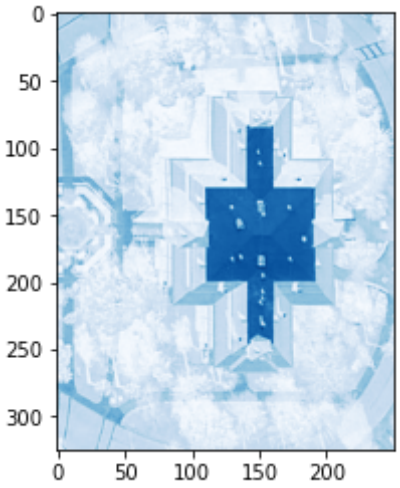
In [27]:
```
# Based off the image above the bounds are
r1 = 250
r2 = 575
c1 = 350
c2 = 600


DataRed_om=Sheet_Red.iloc[r1:r2,c1:c2]
DataGreen_om=Sheet_Green.iloc[r1:r2,c1:c2]
DataBlue_om=Sheet_Blue.iloc[r1:r2,c1:c2]
```

```
# Display 'Red' band
plt.figure()
plt.imshow(DataRed_om,cmap='Reds')

# Display 'Green' band
plt.figure()
plt.imshow(DataGreen_om,cmap='Greens')

# Display 'Blue' band
plt.figure()
plt.imshow(DataBlue_om,cmap='Blues')
```

Out[27]: <matplotlib.image.AxesImage at 0x7ff76a66bd00>





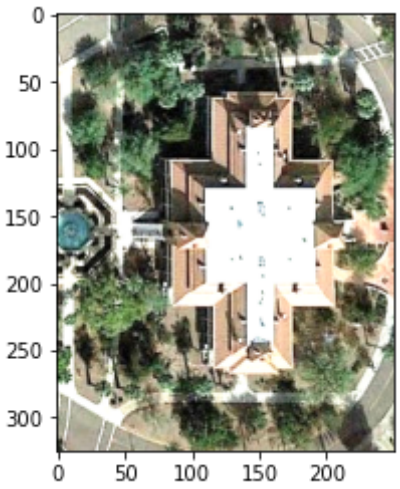


In [30]:
```
DataRed_om = DataRed_om.to_numpy()
DataGreen_om = DataGreen_om.to_numpy()
DataBlue_om = DataBlue_om.to_numpy()
type(DataBlue_om)
```

Out[30]:
```
numpy.ndarray
```

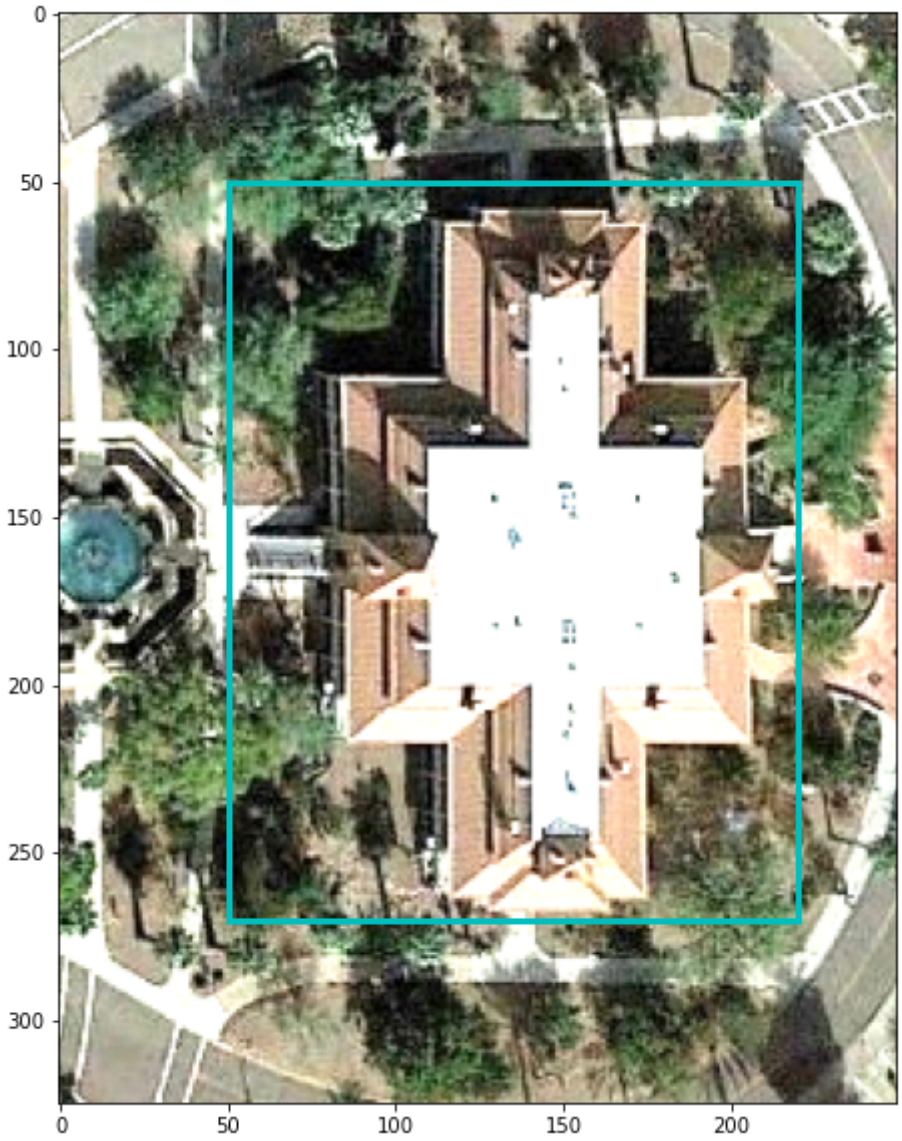
In [31]:
```
# Combine all bands, the Red, Green and Blue data into an RGB model for display
print("Creating RGB Image...")
RGBImage=vip.Image_getRGB(DataRed_om,DataGreen_om,DataBlue_om,250)
# Display RGB Image
plt.figure()
plt.imshow(RGBImage)
print("done")
```

```
Creating RGB Image...
done
```



In [41]:
```
#This library will help youi draw on images
import matplotlib.patches as patches
from PIL import Image

#OldMain = Image.open('./Images/OldMain.jpg')
# Create figure and axes
fig, ax = plt.subplots(figsize=(20,10))

# Display the image
```

```
ax.imshow(RGBImage)

# Create a Rectangle patch
rect = patches.Rectangle((50, 50), 170, 220, linewidth=3, edgecolor='c', facecc


# Add the patch to the Axes
ax.add_patch(rect)

plt.show()
```



--- Exercise 3 ---

In this exercise:

a) Compute the radius for a geosynchronus/geostationary satellite

c) given

- G
- m
- t

In [42]:
```
# load library modules
import os
import math
```

In [43]:
```
# gravitational constant (m3 kg-1 g-2)
G=6.674E-11
#Earth Mass (kg)
M1=5.972E24
# sidereal day (seconds)
T=86164

a=  T*T*G*M1 / (4*math.pi*math.pi)
r= math.pow(a,1/3)
```

In [45]:
```
print(" Radius = ",r, "m = ", r/1000, "km" )

# *** To complete: ***
# Can adjust to AGL - Above Ground - this requires finding the Earth Radius
Earth_Radius_m = 6378100
Earth_Radius_km  = 6378.1
#display a message to know the program ended
print("AGL Adjusted Radius = ",r-Earth_Radius_m, "m = ", r/1000 -Earth_Radius_k
print("program ended.")
```

```
 Radius =  42163111.82545868 m =  42163.11182545868 km
AGL Adjusted Radius =  35785011.82545868 m =  35785.011825458685 km
program ended.
```

Note from Student (Danielle):

It looks like everything is computer for exercise 3 in this lab but let me know if I missed something. Thanks!

- Danielle T