### 

### **Automatic Land-Cover Change Detection Using Python**

**Jimmy Chen:**

Project Lead, Change detection utilizing OpenCV3 algorithm

**Amreen Khureshi:**

Image Classification, Documentation

**Daniel Wanke:**

Documentation, Data organization and collection, Image Classification

**Lauren Hammond:**

Change detection utilizing Principal Component Analysis & K-Means algorithm, Gantt Scheduler

**Maura Antonio:**

Image Classification, Documentation

**Customer:**

**Lei Zou**

Professor

9/19 Discussed the overall trajectory of the project, addressed concerns and approaches

10/24 Asked for advice on how to divide up the work for the group members

11/22 Cleared up classification tasks and other minor details about the project

**Abstract**

This project aims to gain experience with using Python to solve GIS and remote sensing related problems. This project seeks to compare two different Python algorithms regarding change detection between images: OpenCV and PCA/K-Means. Imagery to compare OpenCV and PCA was acquired through non-profit satellite imagery organizations and news articles. The OpenCV algorithm produced a change map indicating changes that had occurred between two classified images. The PCA/K-Means algorithm produced a grayscale map indicating differences between two classified images.

**Introduction (Individual - Lauren Hammond)**

Our group set out to automate the change detection of satellite imagery using Python. The effects of natural disasters such as Hurricane Harvey can clearly be seen in imagery taken before and after the event (Kramer & Sams, 2017), and served as a motivating factor for this project. In examining the differences visually, we were interested in digitizing these changes with image interpretation and statistical methods. A key change we wanted to focus on was the amount and location of flooding in the after image, caused by Harvey. This project has an impact on modern disaster studies and can show areas of serious and prolonged flooding. It can be applied to other land cover change analyses and many other applications in GIS and geography. The original outcome of the project was to compare the results of two Python change detection algorithms- OpenCV and PCA/K-Means. The outcome shifted to focus on the change detection maps produced by the completed OpenCV algorithm, with hopes for PCA/K-Means code to one day get to the same level of functionality. Specific goals included the classification of before and after imagery into land cover type categories, production of a color change map showing the differences between the two, and a statistical measure of percent change for each type. To complete this project, we first collected satellite imagery from around the Houston area showing clear flooding from Hurricane Harvey; the imagery was then pre-processed for standardization. The imagery was classified and analyzed using the OpenCV algorithm for production of a land cover change map. The PCA/K-Means algorithm produced a difference image showing overall change in grayscale. Specific deliverables for this project include: the OpenCV algorithm, the in-progress PCA/K-Means algorithm, a grayscale difference image, and the full-color change map.

### **Project Novelty**

Change detection in Geographic Information Science is a method of understanding how a given area can change between two or more time periods. Change detection can be used to understand ecological changes, or track modifications after natural disasters. Change detection involves comparing changes between aerial photographs taken over different time period that cover the exact same geographic area. Previously, change detection can be accessed through ArcGIS, QGIS or ENVI (third party software). This project seeks to create the same change detection program using native Python coding.

### **Approach**

To accomplish this project, the first step was to find algorithms that could detect change between two images of the same geographic location. After finding algorithms that could be used to detect change between two images of the same geographic location, OpenCV (“Opencv Detect Changes Between Two Photos Taken By Different Time, “2017) and PCA/K-Means (Kumar, 2017, “Unsupervised”) were chosen as the best algorithms to detect change between two images of the same geographic location using Python.

The second step was to find a pair of images of the same geographic location that visually showed change to allow for confirmation that the algorithm was successful in detecting changes between two images of the same geographic location. The images this project focused on were areas along the Texas coast that had been flooded or had been hit by a hurricane in the last decade. The images that were chosen for this project were before and after images of the effects of hurricane Harvey over Brookshire, Rosenberg, and Wharton counties in Texas. Below is a pair of images that was used.



Fig. 1. Satellite Imagery of Rosenberg, TX before and after Hurricane Harvey (DigitalGlobal)

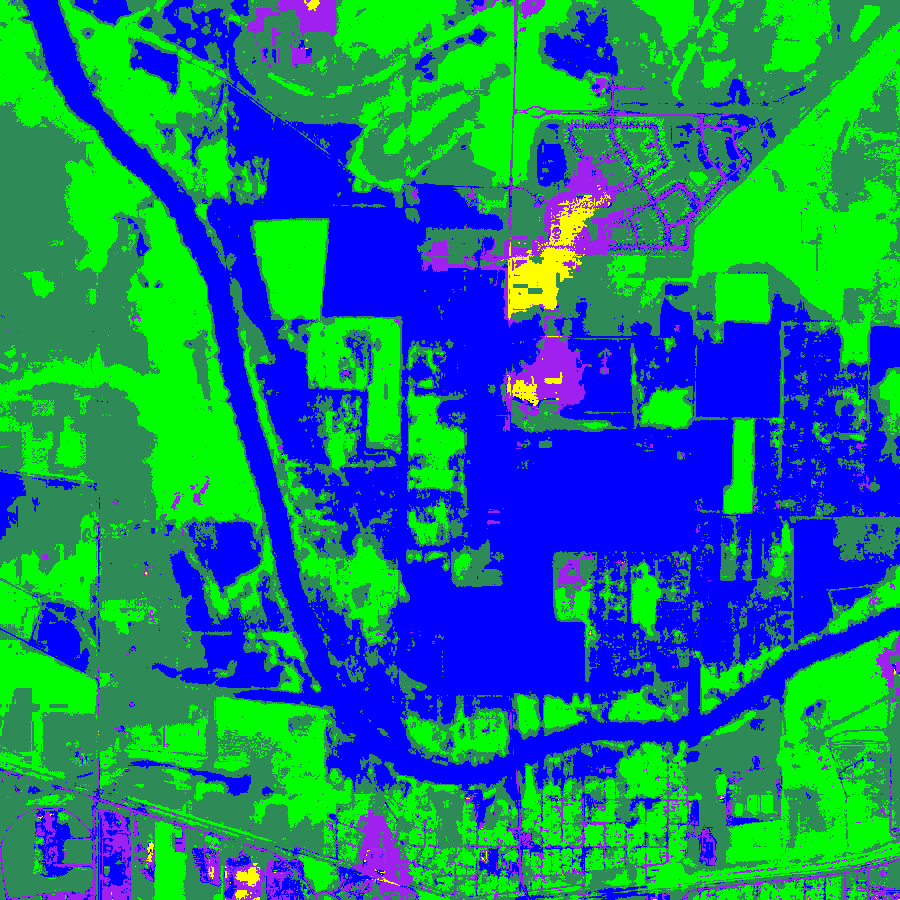
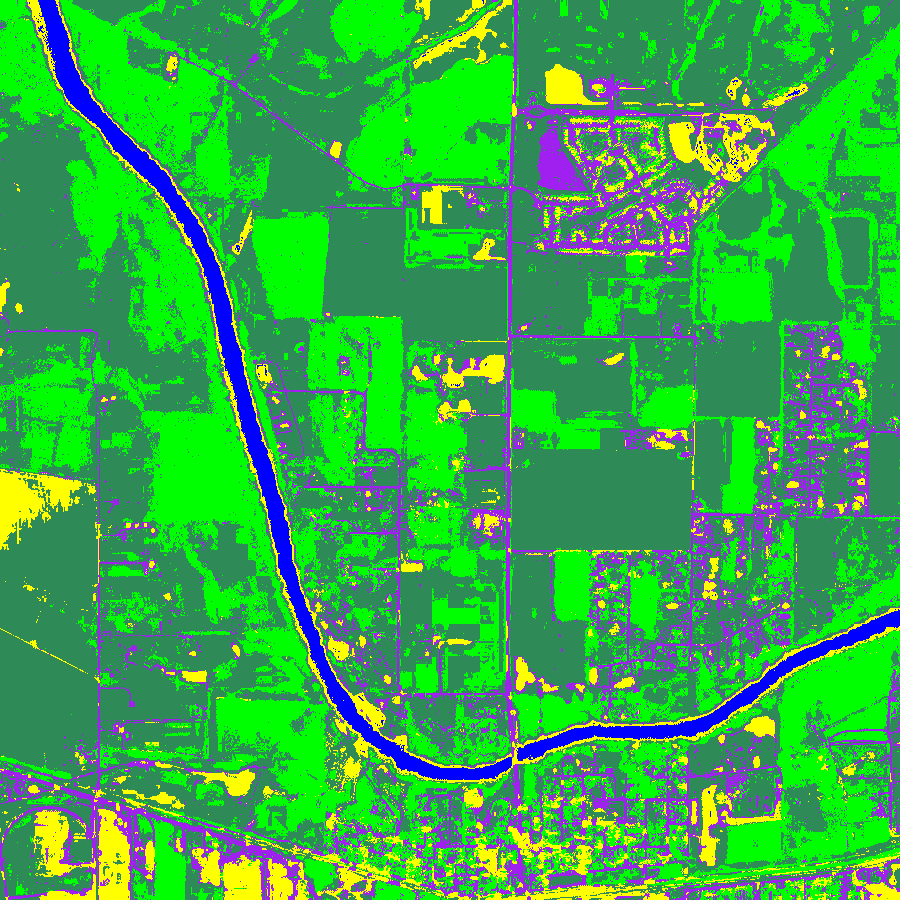
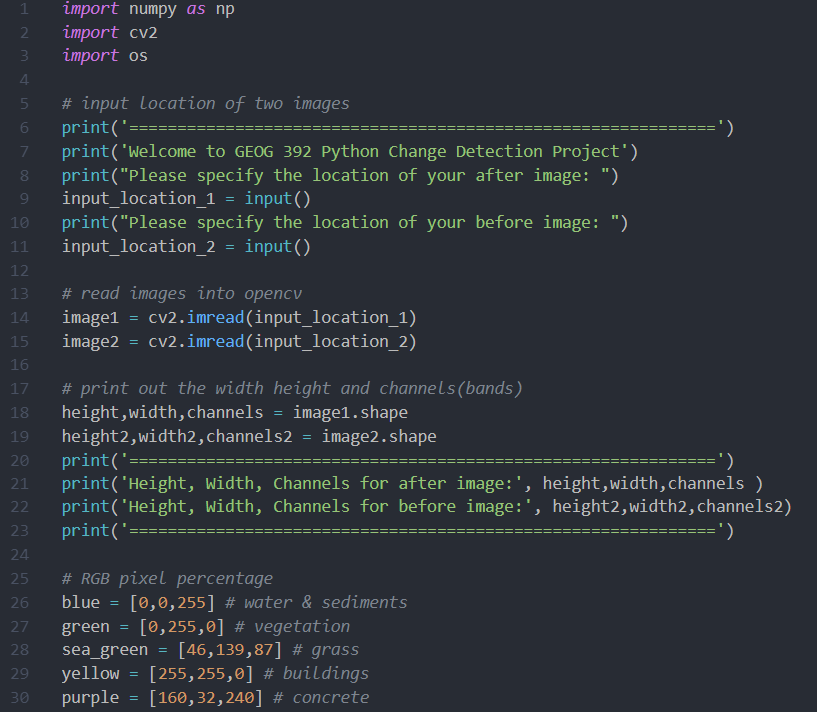


Fig. 2. Classified Map using ENVI of Rosenberg, TX before and after Hurricane Harvey

The third step was to determine if the chosen algorithms could detect changes in the selected images. The first algorithm to be examined was OpenCV. OpenCV is a great package for image change detection since it comes with many preloaded functions/methods that can process images easily. Below is the code explained step by step.

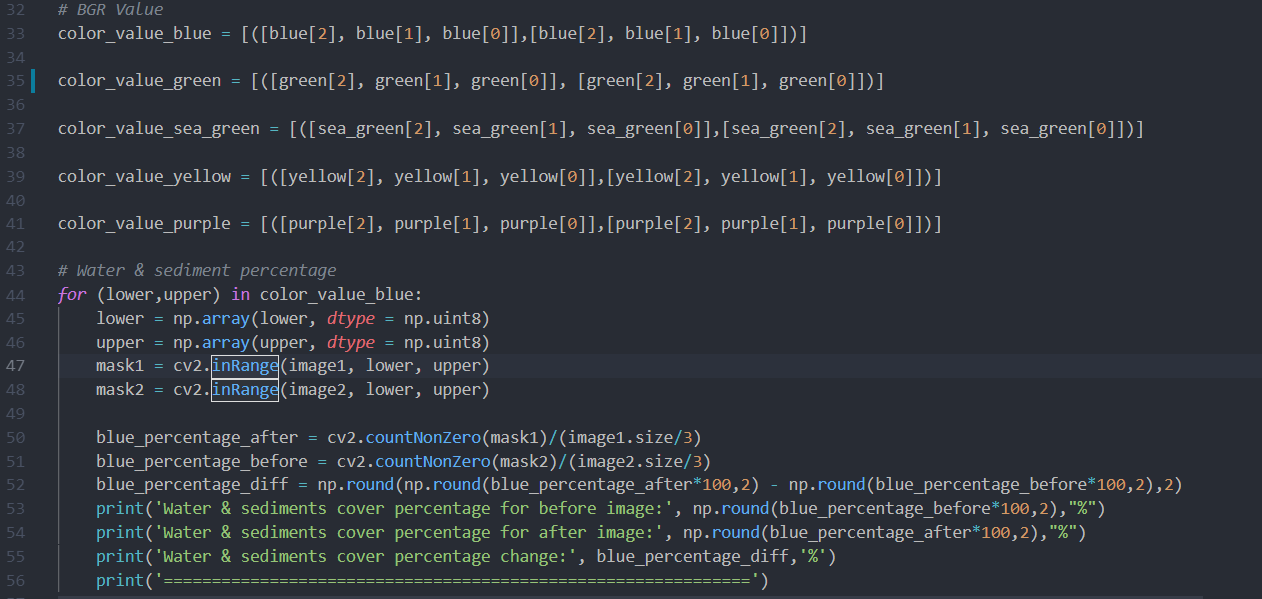


Line 1-3: Import numpy, cv2, and os packages

Line 6-14: Takes input (pre-classified images) from the user, both after and before images. Declare variable, store both images in image1 and image2.

Line 18-23: Print height, width and bands of the images to make sure they match

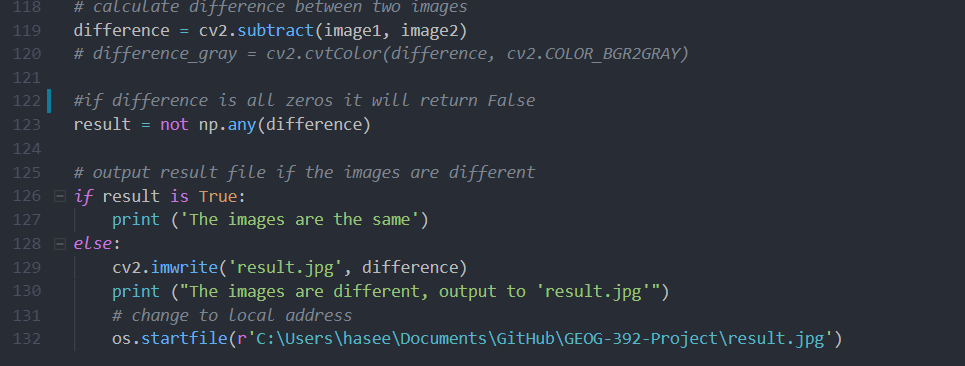
Line 26-30: Declare RGB lists for all five classes



Line 32-41: Rewrite RGB lists into a BGR list for OpenCV processes later

Line 44-56: For loop for color pixel detection on the pre-classified map

* Using CV2 methods such as inRange, countNonZero to calculate the pixel numbers
* There are 5 total for loops for each of the RGB (class) values.
* Calculation of the percentage of pixel cover on the map (line 50-51)
* \* initially the program included a lower and upper limit for each of the for loop, if the pixel color is not exactly the declared RGB value, the program will still count it. This was removed in the end because there were no differences in the 5 color values.



Line 119: use CV2 method to subtract each pixel value (after image - before image)

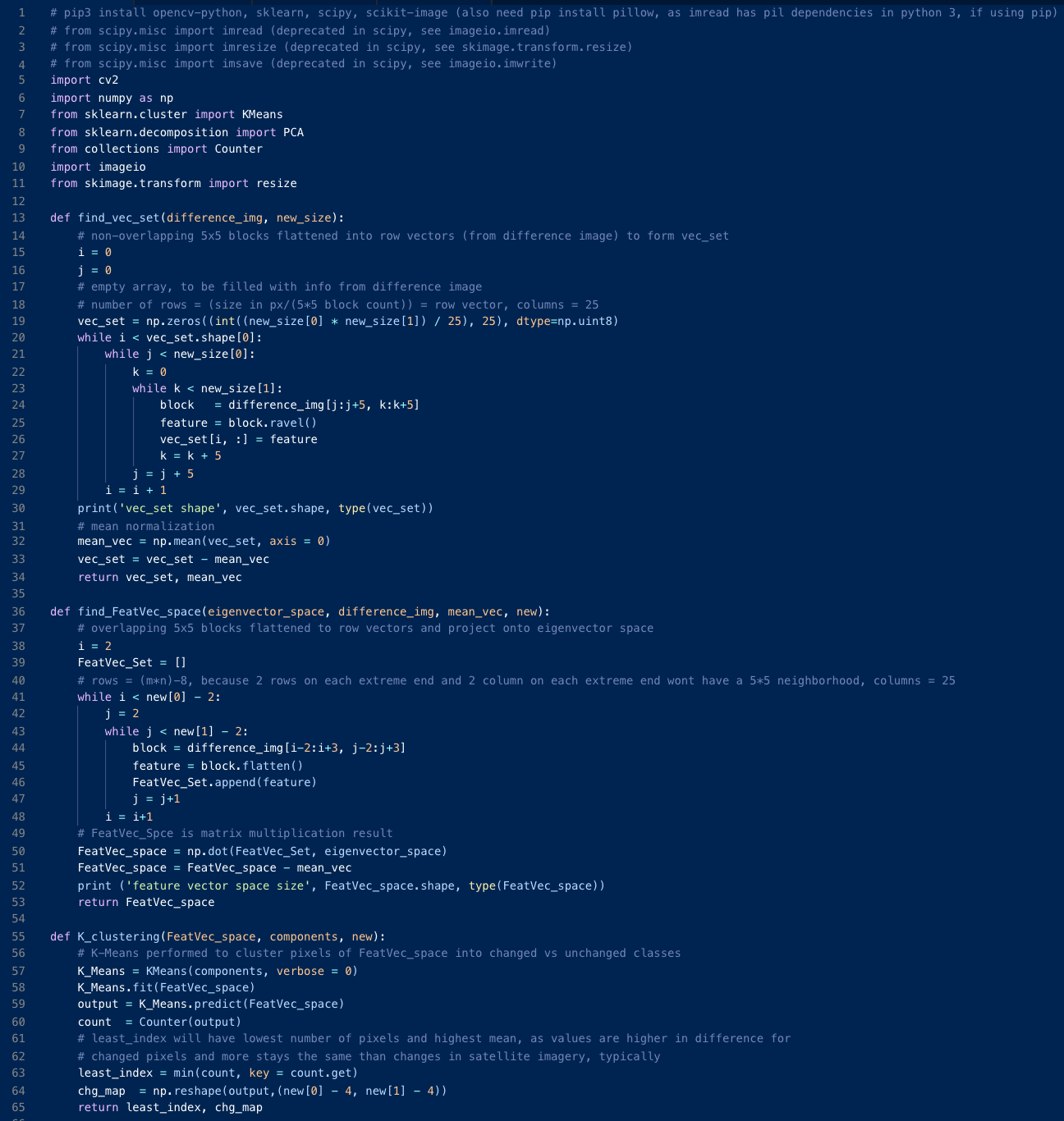
Line 123-130: if difference are all zeroes than the program will return false

* Result will be a bool value
* Print final result whether it changed or not

Line 132: Automatically open image file using os.startfile method

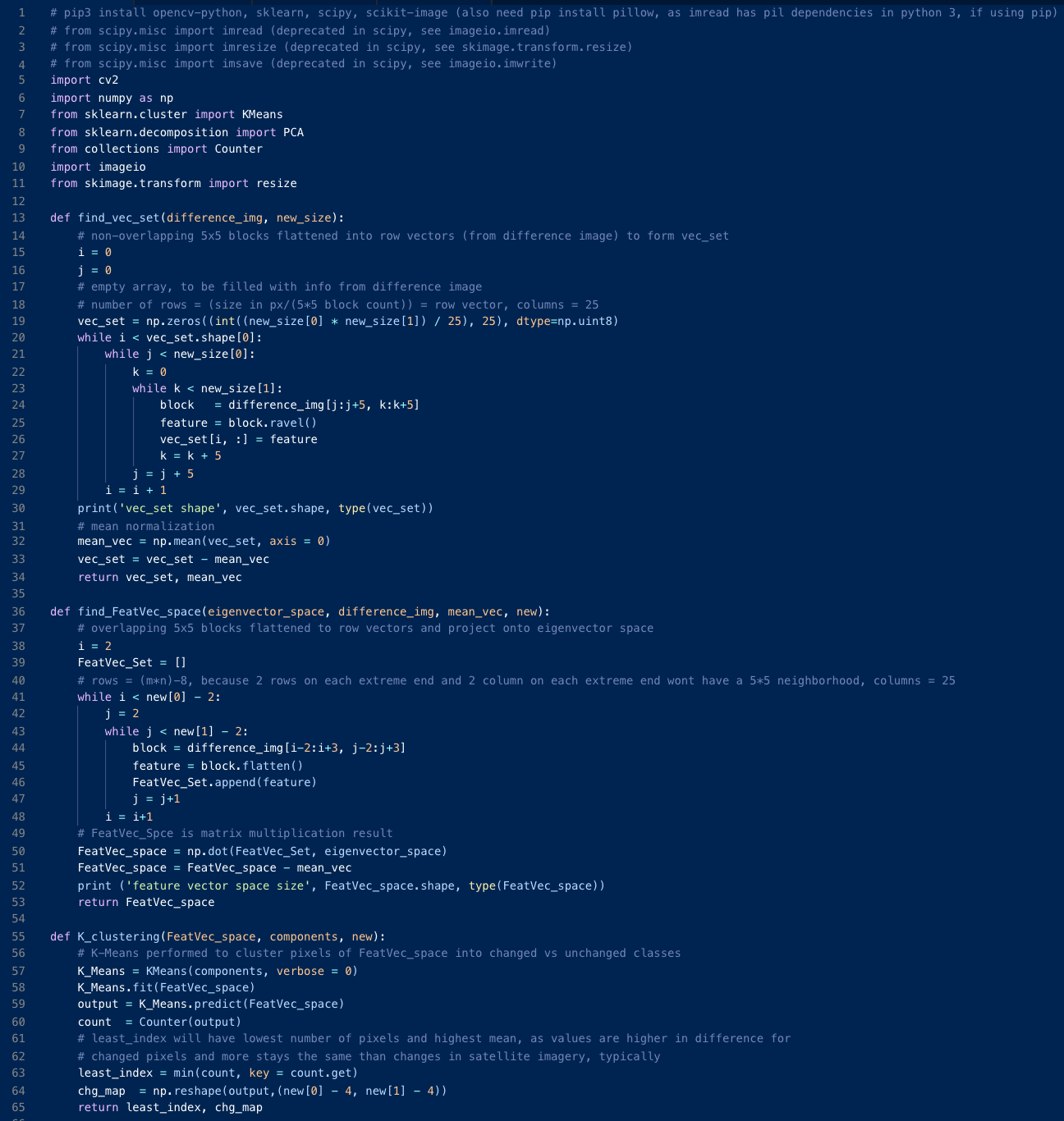
* This will open the change map

The PCA/K-Means algorithm required significant modifications to properly detect changes present in the selected images. The goal of this algorithm (Kumar, 2017, “Unsupervised”) is to perform a Principal Component Analysis on the difference of 2 images, where the eigenvector with the highest eigenvalue and co-variance is the principal component. A K-means cluster is performed to group the difference image pixels into changed and unchanged classes. These classed are then visualized in black and white, respectively, on the final change map. Details are explained below.

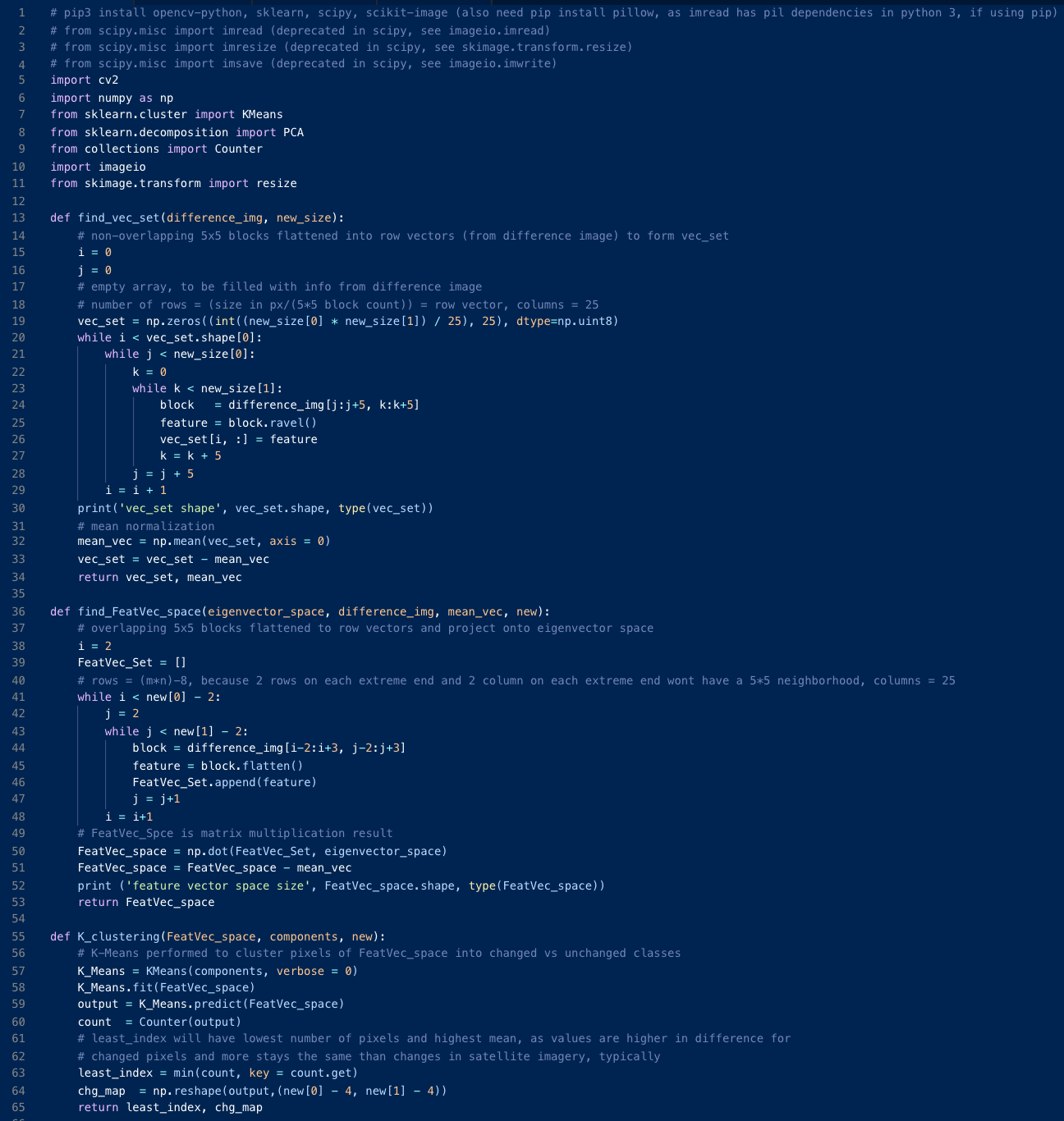


Line 1: Install necessary packages.

Lines 5-11: Import necessary modules, favoring the new image processing tools over their scipy equivalents, which have been deprecated with backwards incompatibility in the latest version.

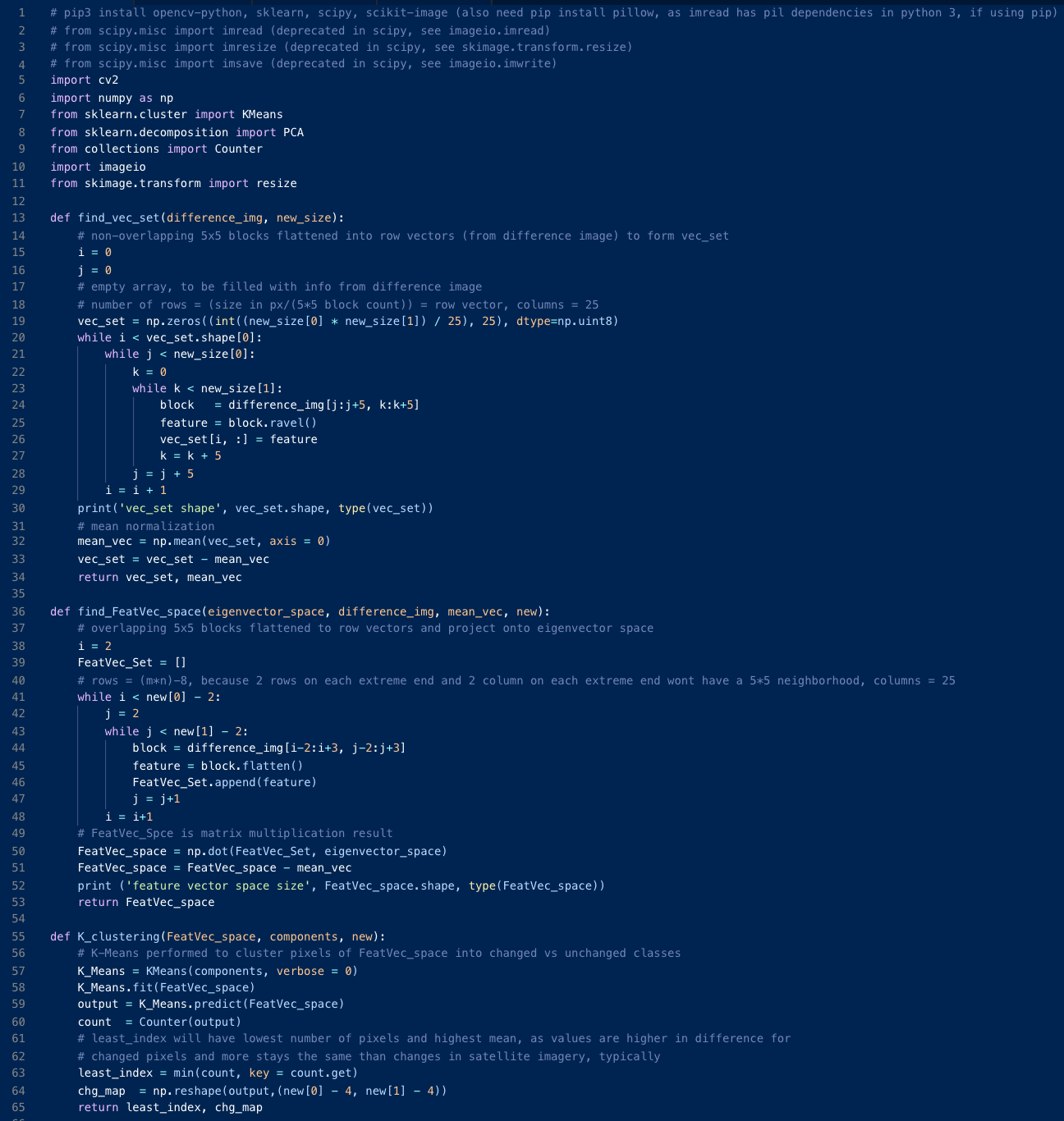


Lines 13-34: Define how to find the Vector Set by flattening rows of non-overlapping 5x5 neighborhoods from the difference image, each becomes a row of 25 dimensions



Lines 36-53: Define how to find the Feature Vector Space by flattening 5x5 overlapping block neighborhoods from the difference image, each becoming a row of 25 dimensions

* Project that Feature Vector Space onto the Eigenvector Space, which will sort the vectors so that their variance is emphasized



Lines 55-65: Define the K-Means clustering so that changed pixels and unchanged pixels are sorted into 2 different classes and change map is produced

* The class with the lower count and the higher mean represents the change pixels





Lines 67-116: Defines how to find PCA/K-means for 2 change images

* Reads in the imagery
* Constructs the difference image as the absolute difference between image 1 and image 2, per pixel
* Runs PCA on the vec\_set constructed from the difference image to sort the eigenvectors and determine the principal component (the eigenvector with the highest co-variance)
* Finds the feature vector space as defined above.
* Change map is generated as defined above.
  + Color ramp is defined: change pixels are white, and unchanged are black.
  + Change map is made by displaying the color for each classed pixel, moved along by a kernel.
* The outer border of change pixels is eroded as to not over-estimate the data in the clean change map.



Lines 118-122: find\_PCAKmeans is run in entirety on the 2 selected images

A challenge that was encountered during this project was designing a working OpenCV and PCA/K-Means algorithm that could accurately detect and display a map showing the changes between two images. A different challenge that this project encountered was image collection. Image collection presented a challenge as the satellite imagery industry is dominated by private organizations. This poses a challenge since private organizations are not usually non-profit and this project had insufficient funds to purchase satellite imagery. Another challenge with image collection was finding a pair of images with the same geographic location and with enough change to notice visually.

During the development of OpenCV change detection algorithm, there were a few difficulties. In the beginning, the method was a simple color change detection, but when a classification method (Pixel Color value detection) was added, a lot more information can be derived from this method.

In regards to the PCA/K-Means algorithm, multiple issues presented while analyzing RGB, 3-channel change imagery. First, the sourced code (Kumar, 2017, “Change”) was written in Python 2.7 and needed to be converted to Python 3 syntax for working longevity. Print statements were written, and more were added so we could see what point the code ran up to before encountering an issue. In changing Python versions, required packages needed to be updated to the newest versions. In the Scipy package, important image processing modules and functions were written in the code, but they have been deprecated or phased out in the latest package update and did not run at all. Alternate versions were found in a package called Imageio (“Transitioning from Scipy's imread,” 2018). There was an array mismatch when projecting the feature vector space into the eigenvector space, as the feature vector space produced 3 sets of 25 dimension arrays (one for each color channel) and would not fit into the 25 dimension eigenvector space. This issue was resolved by reading in one channel, grayscale imagery for the present state of the code. In the future, re-working the eigenvector array to account for three channels will be done. After reading in grayscale imagery, there is still an issue in calculating the final change map. There is a float object somewhere an integer is required, and even after a round of debugging the culprit was not found (many were changed and did not affect the error.



Fig.3. Grayscale Classified Imagery Read in by the Current PCA/K-Means Algorithm

For this project, each member took on a different responsibility. As the leader, Jimmy was responsible for: communicating with Dr. Zou, assigning tasks to the rest of team members, and writing the OpenCV change detection algorithm. Amreen and Daniel were responsible for data documentation. Maura, Amreen, and Daniel were responsible for image classification. Daniel and Maura were in charge of image collection and post-processing. Lauren was responsible for modifying the PCA/K-Means algorithm and creating the Gantt chart.

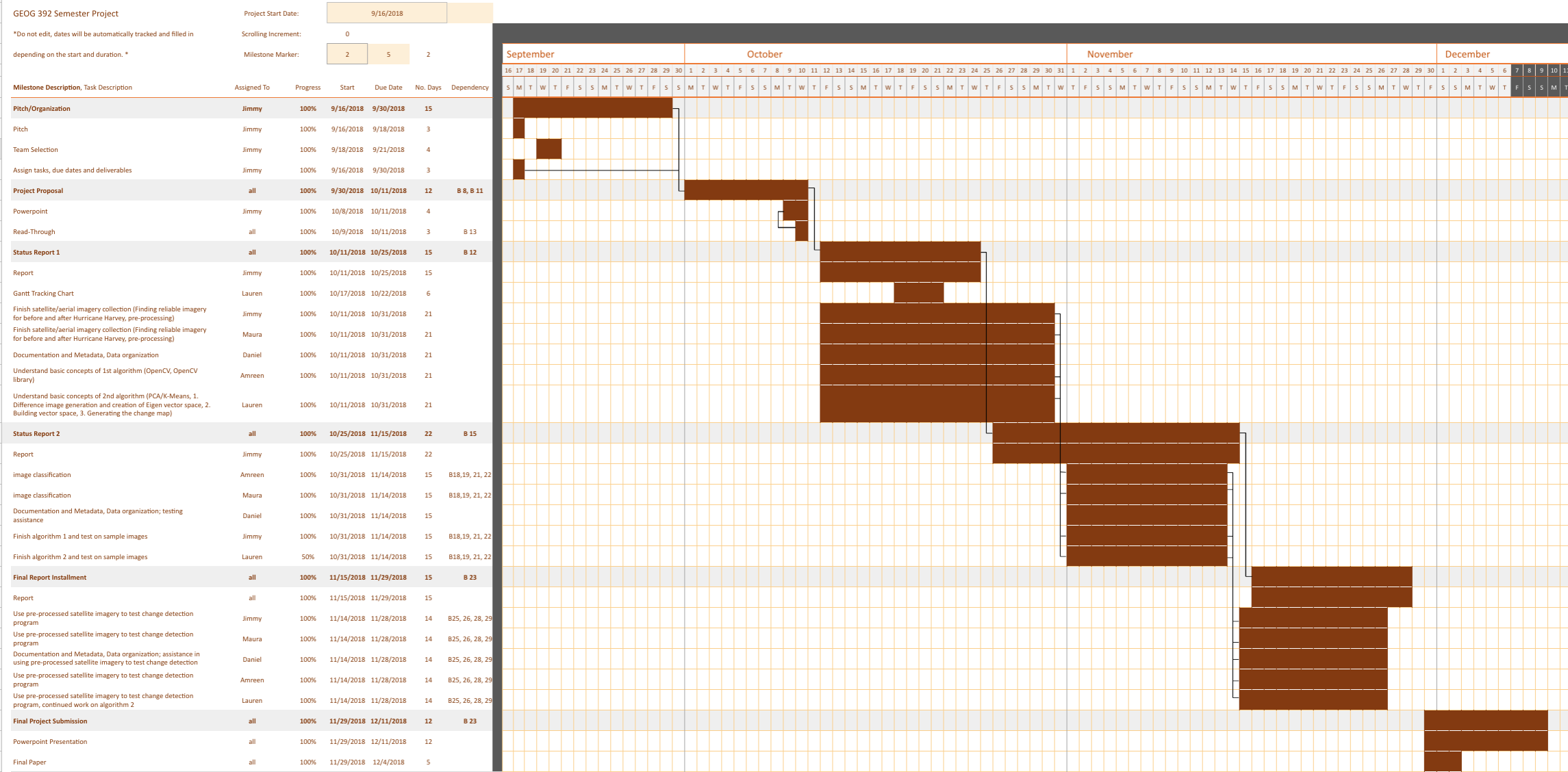


Fig. 4. Gantt Chart Showing Updated Responsibilities, Dependencies, and Progress Tracking

**Project Outcomes & Results**

The main outcomes of this project were a better understanding of how to use Python to solve problems related to GIS and remote sensing issues. This project has achieved most of the results that were expected. This project has successfully created a code that will detect change between two images of the same geographic location utilizing the OpenCV algorithm. Another result that was achieved was creating a code that produced a map showing the change between two images that an algorithm found. The PCA/K-Means algorithm was partially achieved. It was able to detect the difference between two images, and generate a map showing those differences. The PCA/K-Means code runs properly until Line 96, computing of K-Means. Both the color (OpenCV) change map and grayscale (PCA/K-Means) difference image are included below. Classification of the color change map is also included.

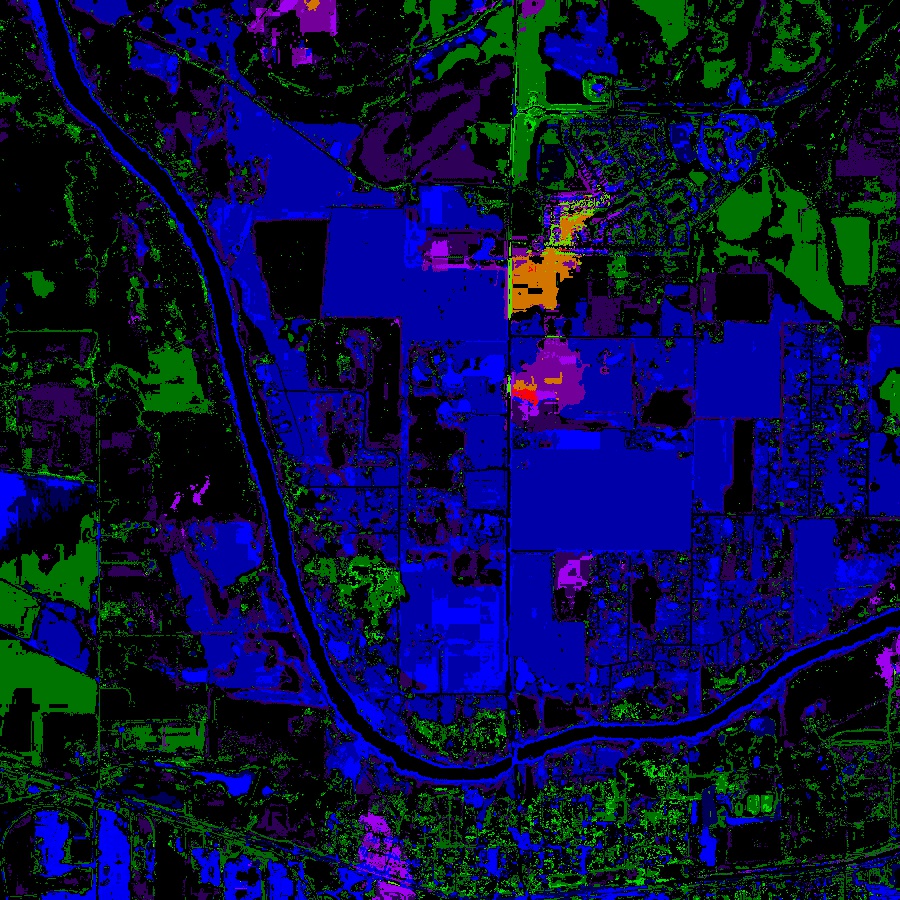


Fig. 5. OpenCV Change Detection Map and Legends



Fig. 6. Grayscale Difference Map Generated with PCA/K-Means Code

The output of this project includes one fully functioning automatic change detection code in the form of OpenCV script. It could be easily worked into a toolbox script for implementation in ArcGIS Pro. To properly run, it requires two different images of the same extent and resolution which have been classified by land cover type. The PCA/K-means code, while part of the project and an originally a goal output, functions partially. It requires grayscale classified imagery as opposed to full color/RGB imagery, and produces an absolute difference map. It has the potential to add greatly to this project in its final form by producing a full color, classified change map. and will continue to be worked on. As a system, the results from the OpenCV and the PCA/K-means scripts will be compared to further improve both change detection techniques.

### **Project Discussion (individual)**

The goal of the project was to produce a fully functioning code for automatic change detection along with change maps showing impacts of Hurricane Harvey. Because this was accomplished, we met the needs of the project. We adapted the project after consultation with our customer and decided to classify the imagery using ENVI so that type change was more clear. Moderate technological readiness is needed to run this code; the user must be familiar with a coding interface such as VSCode, installing Python packages, and file path modification. Minimal input is required other than the image file locations. We decided together to give our deliverable code a Technological Readiness Level of 5, as development is still possible and could improve the strength of the analysis. We would personally like to develop the OpenCV code into a ArcGIS Pro toolbox for ease of operability. Documentation includes a project GitHub repository (<https://github.tamu.edu/Jimmy-Chen-97/Land-Cover-Change-Detection>) with a README.md file which outlines the basics of running the code. In the future, we would like to continue development of the PCA/K-Means code so it can detect change in full color RGB imagery.

### 

### **References**

Ibrahimokdadov. (2016, January 27). OpenCV3 Tutorials 9: Comparing Images and Displaying Difference. [Video file]. Retrieved from <https://www.youtube.com/watch?v=LNzC4NYYWdg>

Kramer, M., & Sams, L. (2017, August 31). Before and After Photos: Harvey's Devastating Floods are Clearly Visible from Space. Retrieved from https://mashable.com/2017/08/31/satellite-before-after-photos-harvey-flood-texas/

Kumar, A. (2017, November 25). Change-Detection-in-Satellite-Imagery. Retrieved from <https://github.com/abhijeet3922/Change-Detection-in-Satellite-Imagery>

Kumar, A. (2017, November 25). Unsupervised Change Detection in Multi-Temporal Satellite Images using PCA & K-Means : Python code. Retrieved from <https://appliedmachinelearning.blog/2017/11/25/unsupervised-changed-detection-in-multi-temporal-satellite-images-using-pca-k-means-python-code/>

Opencv Detect Changes Between Two Photos Taken By Different Time. (2017, June 27). Retrieved from <https://stackoverflow.com/questions/44785958/opencv-detect-changes-between-two-photos-taken-by-different-time>

Transitioning from Scipy's imread. (2018, March 8). Retrieved from https://imageio.readthedocs.io/en/stable/scipy.html

Uz, S., Pearce, S., & Prados, A. (2018, August 7). Advanced Webinar: Change Detection for Land Cover Mapping. Retrieved from <https://arset.gsfc.nasa.gov/land/webinars/adv-change18>

Wei, H. (2016, March 8). Change Detection Using the Python Raster Function. In Proceedings of the 2016 Esri Developer Summit. Palm Springs, CA. ESRI. Retrieved from <http://proceedings.esri.com/library/userconf/devsummit16/papers/dev_int_89.pdf>