# Plan

The goal of the project is to take a photo as input and label the mountain peaks that are present in the image. GPS and Image data is scraped from the EXIF information which gives the team important information such as latitude, longitude, and heading, as well as focal length, sensor dimensions, camera orientation, alongside other important features. Given this data, the angle of view (AOV) and the image’s homography (relative to the topographic map) can be calculated.

Using data scraped from KML files, the team is able to query a created database for the nearest mountain peaks. To expand the capability of the system beyond the KML data, the Overpass API is used to query for additional nearby peaks. Using Google’s Elevation API, the elevation between two locations can be found and we are able to tell what mountain peaks are visible in the line of sight from the viewer.

To determine the relative position of each mountain peak within an image and match the correct name to the correct pixel location we are using the homography between the topographic map frame and the camera frame to project the map into the camera’s view. We can then see approximate pixel locations of mountain peaks within the frame. Due to the lack of precision in the camera’s orientation and compass data the accuracy of reported pixel locations of mountain peaks will vary using this method. Instead, this data will be used to match peaks located visually in the image to peaks that should be seen in (and around) an image’s FoV using their estimated relative positions from the perspective of the camera.

For quality annotations, our program needs to be able to accurately identify the pixel location of a mountain peak within an image. Our initial approach involved finding edges and corners in the image and then identifying the peaks. Unfortunately, after many hours of trial and error we were unable to produce satisfying results. To rectify our peak identification problem, the image is sent through a Faster RCNN architecture with transfer learning from the VGG19 model with image-net weights to place bounding boxes on the mountain peaks within the photo.

The team decided on transfer learning as a starting point due to the lack of pre-labeled mountain-peak data-sets publicly available, and image-net’s inability to complete this task out of the box. To obtain an adequate amount of training data for this task, we are in the process of creating a dataset of localized mountain peaks. Using a list of query terms, the team has scraped several thousands of mountain peak images and is in the process of hand-labeling the images with bounding boxes on each image's visible peaks. To get more data and add to the robustness of our model, image augmentation classes that warp both the input image and the output bounding box location have also been employed. Flips, rotations, translations, scaling, and shearing have been applied across our dataset to increase the size of our dataset. Finally, a resizing class formats all our model’s inputs and bounding boxes into VGG16’s 224x224 default input size.

The final step in the process is to stitch each component together to create a satisfactory final product. Once we have identified the names visible peaks within a given image’s FoV, their relative locations, and the pixel location of each peak, we can then match the theoretically visible peaks to their respective pixel locations and annotate the image. Our final output will be the input image, with the names of each visible mountain peak annotated in the mountain's location.

# Done So Far

Our work thus far has been centered around data collection, formatting, and class structuring to expedite our development process. Various querying functions have also been created to allow us to navigate the large amounts of data involved. We have successfully sourced and combined datasets to create a set of over seven thousand mountain peaks in the state of Colorado and have matched these to a high resolution topographic map.

From here we have been successful in using the EXIF data from iPhone images to generate a homographies between a camera’s frame and the map frame that allows us to project the topographic map onto an image. This projection also allows us to analyze the expected line of sight in an image and identify the peaks most likely to be in this image as well as their relative locations given the camera’s vantage point.This projection takes the curve of the earth surface into account in addition to it coordinate position as it can dramatically affect line-of-sight at these scales.

For peak identification, we have tried and failed at implementing edge/corner detection solutions and thus we have moved onto a machine learning solution. We have implemented and tested pixel wise mountain peak estimation in addition to with a custom CNN solution and VGG16 solution with poor results. Both of these methods have proven to require a large amount of processing power and training data, both of which we are short on. To solve this issue, we have chosen to create our own dataset and use the Faster-RCNN model in tandem initialized with pre-trained weights for the convolutional network from image-net.

We have created a tool that is aiding in our dataset creation by scraping thousands of images from google that we then hand label with bouncing boxes on the mountain peaks. We currently have several thousands of images scraped and slightly under a thousand images labeled. We hope this new custom dataset will allow us to train a robust and reliable model for peak identification.

# Problems

A big question is the definition of a mountain peak. This definition is highly variable but based on peaks, isolation, and prominence to the surrounding area. This means that some data has some peaks labeled and others didn’t classify that peak as a peak. So with the definition being highly subjective, labeling the data is a little subjective. The primary solution was to normalize our labeling technique to have the center of the bounding box be the very tip of the peak. With this method, the threshold of peak detection probability can be tuned to accept mountain ‘peaks’ that are less commonly classified as peaks.

The team’s biggest current challenge has been identifying the peaks in the image. To tackle this, the team split into two groups. The first group is focusing on using the homography, AOV, and the real world points (longitude, latitude, elevation) to project the points into the 2D image (Similar to homework 1). The other group is working on the Faster RCNN architecture which should help to find the peaks in the image. The Faster RCNN method was chosen over simple classification due to the performance of pixel-wise checking and the fact that some mountain peaks aren’t just a single point, but more of an area (such as if the tip of the peak is flatter).

Given the location of the peaks in the image, the identity should be able to be determined by the line of sight of each mountain peak in the database and the order/location the peaks appear in the picture.