CS230

Geographic Point-of-Interest Detection With YOLO
Category: Space Applications

Flynn Dreilinger (flynnd), Grant Regen (gregen), Jesus Meza (jemeza)

Stanford University

1 Project Description and Purpose

Satellites rely on external agents, such as global navigation systems like GPS to determine their positioning and precise orbit determination. These systems provide highly accurate position data for satellites and only require a low cost GPS module, making them a common feature compared to other guidance methods. Due to this heavy dependence on positioning satellites, these systems have become targets for cyber-terrorism[1]. Disrupting GPS satellites or their communication could lead to the failure of critical operations on earth and in space including medical systems, communications, financial markets, and military intelligence.

An alternative, robust approach to navigation and guidance employs analysis of earth images to determine a satellite's orbit. This method, although lacking demonstration, would allow for a decentralized system of control protected from wide external attacks. Furthermore, for deep space missions to rocky objects (such as Mars) that lack global navigation satellite systems (GNSS), this method would enable autonomous orbit determination and future applications to navigation without earth communication.

We propose to prove the use of deep neural networks to allow low earth orbiting satellites to determine their orbit based solely off imagery, time-stamps, and camera specifications. We begin here with our convolutional neural network (CNN) step of image matching from standard satellite data to allow an image to be autonomously matched with another image that has gridded coordinate information interpolated using the GDAL python module [2]. Once matched, classical algorithms such as FAST, SURF, and ORB can accurately place a given satellite's image in reference to a standard set of images that have coordinate data. With a satellite's internal attitude information and a known coordinate value, an orbit can be classically determined with an accuracy largely determined by the coordinate accuracy.

1.1 Past Orbital Determination Efforts

Previous approaches to satellite attitude determination relied on feature detection algorithms such as FAST and SURF paired with algorithms from the RANSAC family for elimination of incorrect features [3]. Images from satellite reference data sets are usually processed with FAST and compared to a base map. Then, conventional methods of attitude and determination are applied. Kouyama et. al. 2017[4] demonstrated 0.02 degree accuracy; however, orbit information is assumed. Sreeenivas 2017[5] demonstrates hybrid invariant combination of surf and brisk that is better than previous methods. They do not include attitude or orbit determination in their research. In addition, Wong et. al. 2015[6] demonstrates slightly different "ASIFT-based local registration method for satellite industry." We diverge from these past efforts by treating image matching not as a feature comparison problem, but a feature identification problem. Instead of focusing on matching all images with a very large basemap, we instead choose a specific subset of highly identifiable earth features. This method allows for a much smaller reference data set that can be stored on storage limited satellites.

While we found several papers[7,8] that incorporated deep learning with satellite imagery, we first focused on a single paper that seemed to best follow our intended methodology of image matching as an initial step toward orbit determination. Multi-Temporal Remote Sensing Image Registration Using Deep Convolutional Features showcases the use of deep convolutional networks in order to make "multi scale feature descriptor[s]" and a point set registration that can be used to register two different images[9]. While we first tried this method due to its superior matching capabilities compared to classical ORB, our results with specifically satellite imagery from varying scales and seasons remained far worse than conventional satellite matching. Due to this failed first attempt, we drastically changed our method to a YOLO feature recognition based method described in this paper.

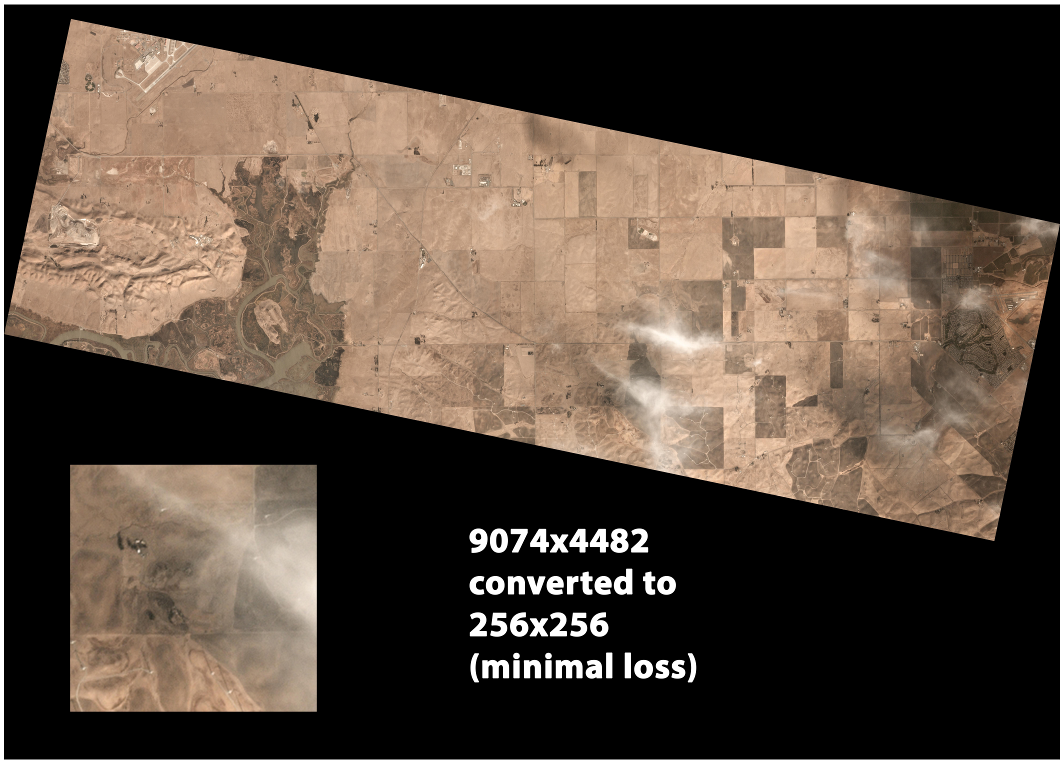

Figure 1: Example of normalized, cropped PlanetScope image of Bay Area; we use GDAL to preserve local coordinate values and Lanczos desampling to decrease resolution to 256x256 while preserving contrast features.

2 Dataset Creation

The creation of our custom data set remained a significant challenge due to an absence of CNN ready, labeled satellite images. In response, we created a tool specifically to create such data sets. Using Planet Lab's API with access through the Stanford Geospatial Center, we created a custom API script that could pull mass amounts of imagery based on date of capture, image location, and cloud cover. The script can pull images from four different constellations of satellites: PlanetScope, RapidEye, Sentinal-2, and Landsat-8[10]. These dozens of satellites offer extremely varied image scales and colors, allowing for a diverse representation of earth imagery.

However, we soon realized that these images were slanted at different angles with varied amounts of blank space around each scene, which wouldn't work for YOLO data. Additionally, we wanted to crop images to smaller square sub-scenes while still knowing their center and edge coordinates. Thus, we incorporated the GDAL python module to slice images into squares with known interpolated coordinate values with position accuracy equal to the original images. With these sub-images, we implemented an option to only download sub-square images that contained a coordinate point from a defined set of labeled positions. This script then saves these images (or images without the defined points if specified) each with a text file containing all metadata.

With this powerful data pre-processing pipeline, we specifically chose to prove our model on a non-global scale due to our limited project time and lack of labeled data. We downloaded image sets of 25 different features in Colorado that remained visible (but changed drastically) from winter to summer and were all around a few kilometers in dimension to match the scale of all our available satellites. Additionally we downloaded data outside of Colorado as negative images. We labeled over 5000 images with around 2000 images having labeled boxes around our chosen 25 Colorado locations. We chose a split of 90%/10% training/test.

The main pre-processing steps we employed were to downsample and crop the images to sizes of 256x256 pixels to work with our chosen deep learning model. We took our high quality tiff images and cropped them into square regions, discarding all cropped images that had blank areas with alpha channel pixels equal to zero. We downsampled with the Lanczos method[11] available in the PILLOW image module in python. Additionally we stored all metadata information in separate text files as labels and saved images as jpegs to decrease the image size.

3 Model Architecture and Thinking

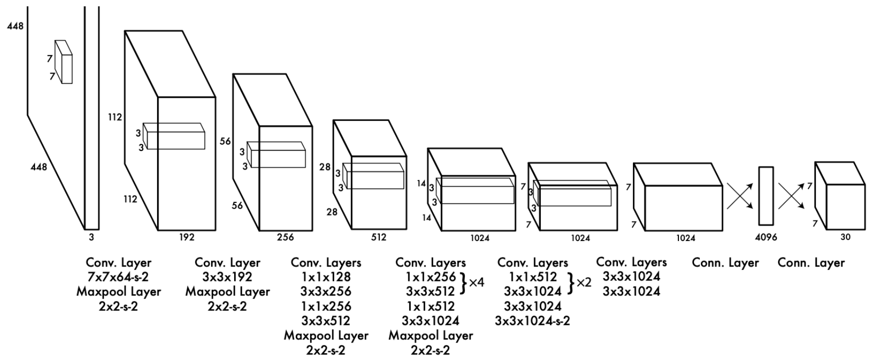

Figure 2: Diagram of YOLO CNN from Redmon et al. 2016. We retrained the YOLO model to create weights custom to our satellite features.

While much of our time remained in gaining the administrative permissions to access satellite data, activating this data, pre-processing it, and labeling it, we spent considerable time testing the model described in Yang et al.[9]. This deep learning network failed to match images well, so we turned methods considerably to a YOLO Version 3[14] (You Only Look Once) CNN, which has proved to locate trained features faster and more accurately than other CNN's for non-satellite applications(Redmon et al. 2016).

After our first model failed, we drastically rethought our solution. Instead of matching images with a global base map, which would take an incredible amount of data and processing time, we thought of matching images with very recognizable surface features such as lakes, cities, and strange geological features. Similar to using YOLO to finding a car or cat, these locations vary slightly in appearance seasonally, with cloud cover, and with varying scale. Our diverse data set allows the YOLO model to recognize each of our 25 locations as unique objects, able to be boxed just like a conventional YOLO object. Due to the comprehensive nature and testing of the YOLO model, we decided not to add additional layers and instead followed the developer's comprehensive instruction with a Darknet implementation.

3.1 Hyperparameters

When choosing hyperparameters the team took into consideration the hardware being used, the amount of data, and past experiences. The learning rate and burn in were chosen largely based on the hardware; since four GPUs were used a lower than normal learning rate was used in order to keep the learning steady. The batch size was chosen to be 64, this choice resulted from our dataset size and past experiences. Other hyperparameters used were based on standard practices for the yolov3 framework.

4 Results and Analysis

Unfortunately, we didn't have enough time to create a large enough labeled data set to create extremely accurate matches, but with the small size of our training set for the 25 features, we saw promising matching. These results proved our solution to the matching problem as a novel solution to matching images to any rocky planetary surface with preexisting imagery.

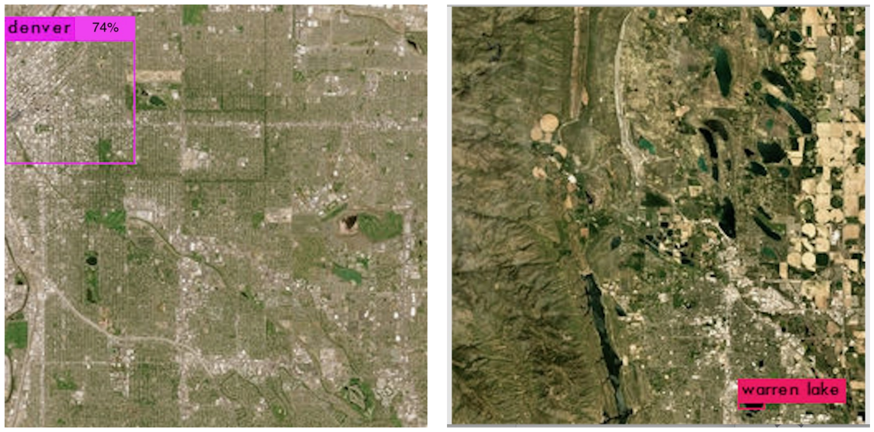

Figure 3: Example location identifications with the YOLO model showing the working identifications with different scales of satellite imagery. The locations had 74 and 48 percent matches respectively.

Specifically, the algorithm detected 90.7 percent of the features in the test set with an average IoU of 40.32 percent and an average recall of 30.47 percent. The most important of these results, however, are the positive detection of over 90 percent of the images with features. We plan to examine to missed images to expand our training set to help better identify these types of missed images. We also want to explore false positives, which we have not fully explored yet.

With these results, while they may be similar to conventional image matching, more training data will significantly aid in better matching. Currently, out locations only have about 35 training images each, which is a tiny number compared to other conventional YOLO object detection models. Additionally, splitting images into a larger variety of resolution scales will allow for a greater range of training scales, which will aid the algorithm in matching different satellite's images with different cameras in different orbits. We look forward to expanding our data sets this summer.

5 Discussion and Next Steps Toward Orbit Determination

In this project, we proved the viability of a new solution to satellite imagery matching with rocky planetary bodies. Specifically, increasing the variety—and more importantly amount—of data will incredibly improve our solution. We started this project specifically in regards to another project all three of us are currently working on. Through the Stanford Space Initiative, we are building and plan to launch our own satellite next year to specifically prove the viability of in-house orbit determination. We hope to improve this YOLO model to work on and test with our personal satellite. For use with our satellite, we will convert the training set to images of the same scale as our satellite—not just a large variety of images—to allow it to work best for our personal system.

To expand this project to orbit determination, we will employ computer vision to determine the altitude of the satellite as well. From this we will use conventional methods to determine the position of the satellite in the geocentric reference frame and use a conventional orbit determination algorithm to determine the orbital elements of the satellite. We plan to propagate orbits with the Simplified General Perturbation Model (SGP4[12], published by NASA and NORAD) and then test against newer images in time series. We will also expand our features to global coverage that our satellite would image.

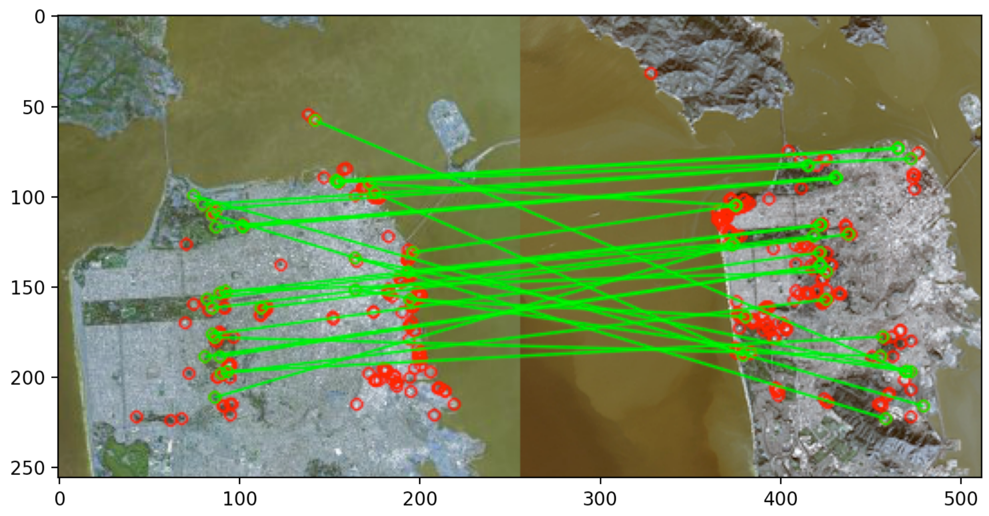

Figure 4: Demonstration of the ORB edge detection algorithm that we adopt to match image locations.

In our next steps we will expand to the next simpler step (that we have already started to develop) of finding the angle of the image and then using a shallow deep learning model to estimate the satellite altitude by comparing images as well as camera specifications. We will use the free ORB algorithm to match YOLO chosen images with their matched location image that contains coordinate information to return precise coordinate values of the center of the image (see ORB figure that shows how the classic algorithm works). We will continue developing and organizing our data as well as adding more labeling in order to develop models necessary for different aspects of our project. We are very excited to show these initial successes of this novel solution, and look forward to sharing the larger model with you next year.

6 Acknowledgments

We would first like to thank Andrew Ng, Kian Katanforoosh, and the entire teaching staff for their tireless efforts to adapt the CS230 class to an online format. In particular, we would like to thank Vineet Kosaraju for guidance in all of our one-on-one meetings, support, and leading fun sections. We would also like to thank Planet Inc. and Stanford libraries for facilitating our acquisition of a dataset.

7 Contributions

Flynn provisioned AWS gpus, and used OpenCV to implement the ORB image matcher. Jesus and Flynn experimented with the VGG16 facial recognition style model, and developed, trained, and refined the YOLO model by tuning the hyperparameters and adjusting layers in the network appropriately. Jesus set up the AWS environment and wrote a script for pulling images from Google Earth Engine. Grant in particular wrote a detailed script to arbitrarily pull images of any location on Earth's surface from 30+ satellites with over 10 years of coverage. This script allows for the creation of specific data sets ready for labeling for CNN satellite models. Everyone downloaded a lot of data, labelled it 2 plus times, and cleaned up the dataset by removing unusable images.

References

[1] McCrea, M. (2018) Experts Warn GPS Vulnerable to Time-Tampering Terrorism. CBS SF Bay Area

[2] Warmerdam F. Rouault E. et al. (2020) GDAL Documentation

[3] Bouchia, R. & Besbes, K. (2010) Automatic remote-sensing image registration using SURF. *International Conference on Machine Vision*

[4] Kouyama, T. et al. (2017) Satellite Attitude Determination and Map Projection Based on Robust Image Matching. *Remote Sensing*

[5] Sreeenivas A. (2017) Satellite Image Co-Registration Based On Hybrid Invariant Local Features. *Journal of Theoretical and Applied Information Technology*

[6] Wang X. et al. (2015) An ASIFT-Based Local Registration Method for Satellite Imagery. *Remote Sensing*

[7] Peng, H. & Bai, X. (2019) Improving Orbit Prediction Accuracy through Supervised Machine Learning. *Advances in Space Research*

[8] Peng, H. & Bai, X. (2018) Artificial Neural Network–Based Machine Learning Approach to Improve Orbit Prediction Accuracy. *Journal of Spacecraft and Rockets*

[9] Yang, Z. Dan T. & Yang Y. (2018) Multi-Temporal Remote Sensing Image Registration Using Deep Convolutional Features. *IEEE*

[10] Planet Corp. (2018) Planet Imagery Product Specifications.

[11] Duchon, C. (1979) Lanczos Filtering in One and Two Dimensions. *Journal of Applied Meteorology and Climatology*

[12] Vallado, D. A. & Crawford, P. (2007) SGP4 Orbit Determination. *American Institute of Aeronautics and Astronautics*

[13] Copernicus Open Access Hub. https://scihub.copernicus.eu/

[14] Redmon, Joseph and Farhadi, Ali. YOLOv3: An Incremental Improvement. *ArXiv*