7-15-2022

DETECTION OF ROTORCRAFT LANDING SITES: AN AI-BASED APPROACH

Abdullah Nasir
Rowan University

DETECTION OF ROTORCRAFT LANDING SITES
AN AI-BASED APPROACH

by

Abdullah Nasir

A Thesis

Submitted to the
Department of Electrical & Computer Engineering
College of Engineering
In partial fulfillment of the requirement
For the degree of
Master of Science in Electrical and Computer Engineering
at
Rowan University
June 15, 2021

Thesis Chair: Ghulam Rasool, Ph.D., Assistant Professor, Department of Electrical and Computer Engineering

Committee Members:
Nidhal C. Bouaynaya, Ph.D., Associate Dean for Research and Graduate Studies, Electrical and Computer Engineering
Charles C. Johnson, Rotorcraft Safety Research Team Lead FAA William J. Hughes Technical Center, FAA

Acknowledgment

This work was supported by the Federal Aviation Administration (FAA) Cooperative Agreement Number 16- G-015 and NSF Award DUE-1610911. This Thesis was also supported by a subaward from Rutgers University, Center for Advanced Infrastructure & Transportation, under Grant no. 69A3551847102 from the U.S. Department of Transportation, Office of the Assistant Secretary for Research and Technology (OST-R). We would also like to thank LZControl for their guidance and assistance with this effort. Via a Cooperative Research and Development Agreement with the FAA, LZControl provided a set of data from their system and subject matter expertise, which provides landing zones for helicopters across the U.S., often complementing the FAA's 5010 database and including locations/sites absent in the FAA's 5010 system.

Abstract

Abdullah Nasir
DETECTION OF ROTORCRAFT LANDING SITES
AN AI-BASED APPROACH
2021-2022
Ghulam Rasool, Ph.D.
Master of Science in Electrical and Computer Engineering

The updated information about the location and type of rotorcraft landing sites is an essential asset for the Federal Aviation Administration (FAA) and the Department of Transportation (DOT). However, acquiring, verifying, and regularly updating information about landing sites is not straightforward. The lack of current and correct information about landing sites is a risk factor in several rotorcraft accidents and incidents. The current FAA database of rotorcraft landing sites contains inaccurate and missing entries due to the manual updating process. There is a need for an accurate and automated validation tool to identify landing sites from satellite imagery. This thesis proposes an AI-based approach to scan large areas using satellite imagery, identify potential landing sites, and validate the FAA's current database. The proposed method uses the object detection technique, one of the well-known computer vision methods used to identify objects of interest from image data. Objection detection techniques are based on the famous convolutional neural networks (CNN) and have achieved state-of-the-art performance. We used FAA's 5010 database to build a satellite imagery dataset that contained manually verified landing sites, including helipads, helistops, helidecks, and helicopter runways. We explored different object detection models, including single-shot detector (SSD), you only look once (YOLO), and various flavors of mask regional CNN (R-CNN). Each model presented a unique accuracy-computational complexity trade-off. After achieving satisfactory performance, we used our selected model to search and scan satellite images downloaded from Google Earth for potential landing sites that may or may not be part of the FAA's database. The model identified 1435 new landing sites and increased FAA's current database by 46%. We also identify methods to improve our proposed model in the future.

Table of Contents

Table of Contents (Continued)

List of Figures

List of Figures (Continued)

Figure Page

List of Figures (Continued)

Chapter 1

Background and Introduction

I-A Problem Formulation and Motivation

The FAA maintains an active database of public and private owned landing sites. This database contains a variety of information regarding landing sites including latitude and longitude coordinates, accuracy of this information is critical for FAA. This database is built on FAA records of 7480 and 5010 forms, these forms are primary reporting tools used by Public and Private entities. However, due to discrepancies in reporting or lack of it in some cases, there is uncertainty about the current database and requires validation. Some site coordinates are known to be off by several hundred meters and even a few miles in some cases. In some cases, Helipads were decommissioned but not reported, hence creating false positives. The FAA is also aware of missing landing sites that are unreported, it is estimated by the FAA that there are upwards of 2,000 missing Hospital Helipads in the US. Lack of reporting is a bigger issue when it comes to private facilities and due to lack of an ongoing audit of those locations, there is a need to develop dynamic tools that can expediently search for unreported Helipad locations. These issues can lead to misleading information for pilots that can result in fuel exhaustion or even fatal accidents.

This thesis proposes a solution to address these issues and meet FAA's requirement of validating and improving the current database, developing an autonomous helipad identification system and to scale the FAA's database. We relied on a database of satellite imagery based on the FAA's 5010 database, our approach was to train a model on manually verified dataset and then build on that to validate the rest. We also wanted to scale the FAA's current database by developing a system for searching for missing Helipads.

Validation of existing helipad coordinates allowed us to identify and remove false positives from current records, it also allowed us to train and validate our model which was then used to search for missing helipads. We verified our approach by finding new helipads and further outlined methods to scale the current database in future works. We also implemented our search algorithm to correct existing coordinates and improve their accuracy.

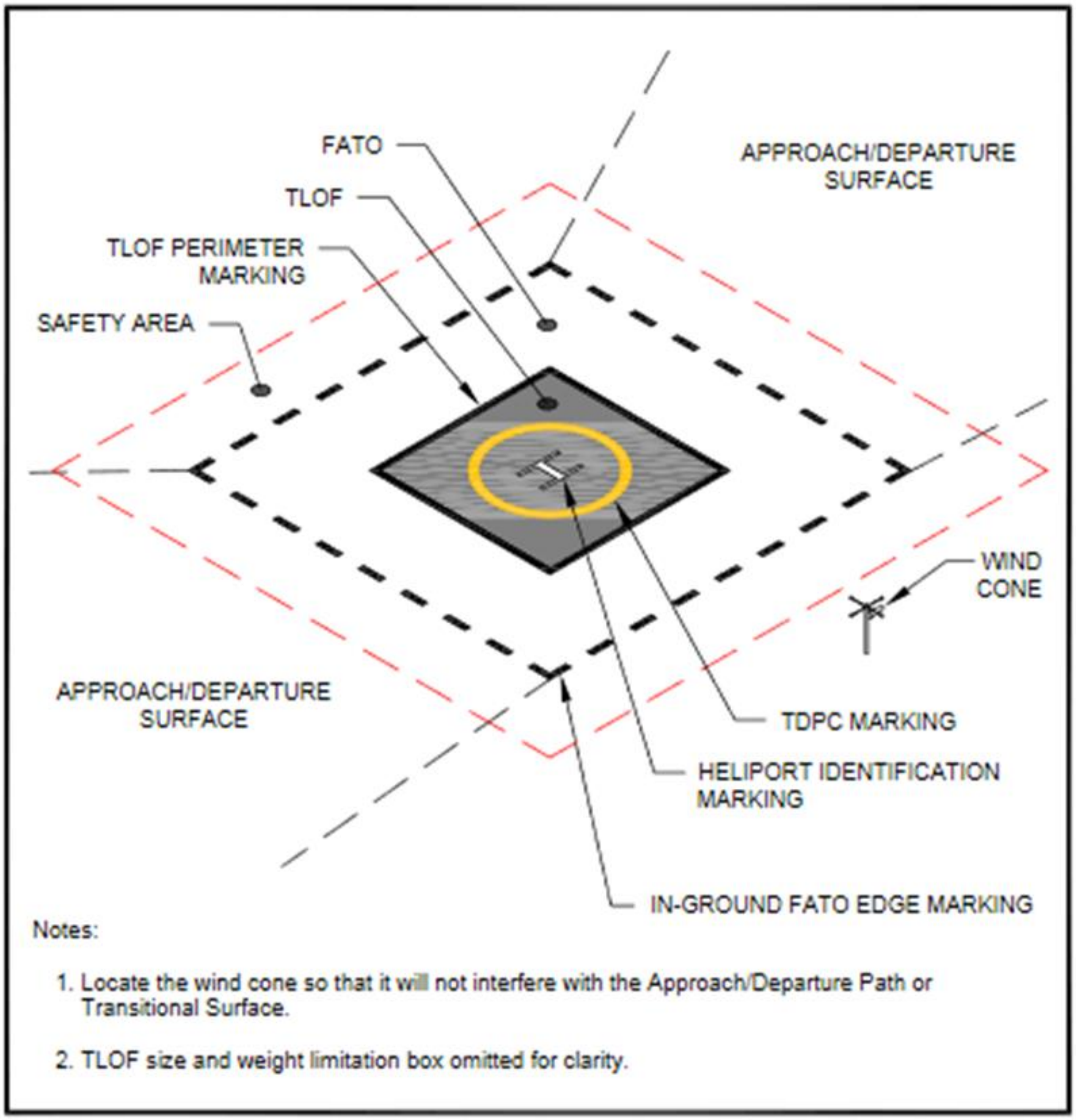
A-1 What are Helipads?

A helipad is a designated area that a helicopter is intended to land, it consists of the Touchdown and Liftoff area (TLOF). The FAA’s Advisory Circular 150/5390-2C [1] defines standards for the construction of helipads.

However, section 203 notes that these guidelines are not set in stone and act as recommendations for Prior Permission Required (PPR) facilities. Section 103 effectively states that the minimum required facilities are a clear area with a wind cone, this leads to a diverse range of helipad sites. Most landing sites follow these standards and use a variant of the markings listed in section 215, which includes a white “H” marking in the middle the marked borders of the landing surface. Of note is that as per section 414, all hospital helipads will also include a cross around the “H”, and section 215 allows for PPR facilities to replace the “H” with another distinctive marking such as a logo.

Figure 1

Structure of a Helipad



Helipads are also a part of some form of landing facility. There are 4 main types of facilities that will contain a helipad. The first type is a heliport, which will have services for helicopters, such as refueling and repairing. The second is a helistop, which is a landing area that offers no services, and allows for clearly marked landing areas without the need to make a heliport. The third is an Emergency Helicopter Landing Facility and this is effectively a helistop that is only used in the event of an emergency. The last type is a helideck, which is a landing facility over the water, such as a boat or oil rig. Along with these facilities we will also consider parking pads and helicopter runways as a helipad for our purposes as they strongly imply that the area is intended for helicopters to land and takeoff at.

Since our primary aim was to validate existing landing sites and search for additional sites, we focused on scaling and recording missing databases instead of a classification approach. Though future works could focus on classifying existing and validated databases into appropriate categories.

A-2 Different Approaches in Helipad Identification

This thesis is a continuation of FAA's partnership with Rowan University to develop Artificial Intelligence solutions, our work is a continuation of previous thesis published on Helipad Detection. [2] That approach focused on using conventional neural network models for binary classification of satellite imagery into Helipad or No Helipad category. It works by producing a matrix function that gives the probability of an image containing a Helipad. Since image classification does not localize the Helipad, it can't be used to search and extract coordinates of missing Helipads. Following up on the image classification approach, we explored other computer vision approaches to meet FAA's requirements for this task. Our previous work references several Helipad related works that have been done previously, focusing on different pattern recognition approaches.[2] We will not discuss those approaches in detail here as we briefly explored conventional computer vision approaches.

One drawback of these approaches is that they are appropriate for specific environments and fail to generalize in complex environments, these approaches work only for specific helipad detection patterns such as “H” only or “Circle” helipad. Hence, you will need multiple algorithms for each specific pattern type, we will briefly discuss one implementation of this approach in a later chapter.

Object Detection approaches in recent years have proven to be fruitful for localization problems and there have been major advancements in development of various object detection models. With having established a benchmark satellite imagery database, this approach was very attractive for our task as it requires a clean labeled dataset that can be used for model training and validation. We will discuss this model-based approach of using “Deep Learning” convolutional neural networks and why we deemed it to be the most suitable computer vision solution for Helipad Detection.

A-3 Object Detection Using Deep Learning

Convolutional Neural Networks (CNNs) are designed to process multi-dimensional arrays such as images. There are 3 layers typically used in CNNs: convolutional layers, pooling layers, and fully connected layers. Convolutional layers consist of filters consisting of learnable weights and are followed by an activation function to introduce non-linearities. By learning values for the weights, the convolutional layer is capable of learning appropriate filters to combine information from previous layers. Pooling layers combine nearby features in an image. Fully connected layers are at the end of the network and will come after the convolutional and pooling layers. A fully connected layer will define multiple neurons, and each neuron will assign a weight to each input and combine the weighted inputs. This is typically followed by a non-linear activation function to add in non-linearities for each layer [7].

There are numerous high performance Machine Learning frameworks that have shown great results over the past decade, these frameworks include tensor flow, Pytorch, darknet, mxnet and darknet etc. We relied on tensor flow, Pytorch and darknet to train different object detection models and compare performance metrics to achieve high accuracy results.

Transfer Learning is a method that has proven to be very effective in Machine Learning tasks as it relies on using pre-trained models and generally only training final or outer layers (fully connected layers). It leverages knowledge (features, weights etc.) of a pre-trained model to train a new model on target dataset, hence improving efficiency of the new model. We utilized transfer learning in our training models which reduced our dataset constraints and allowed us to achieve high accuracy results despite having a small dataset size.

Chapter 2

Dataset: Data, Labeling and Model Preparation

II-A Dataset Acquisition

We used FAA's 5010 dataset as our primary data, this data contained information on recorded helipad landing sites including longitude and latitude. We also used a dataset provided by LZControl, which contained about 200 helipad sites. We used a google maps API to extract satellite imagery on these potential landing sites. FAA's 5010 dataset contained about 5,449 potential landing sites; an effort was made last year to manually verify this dataset but only parts of the dataset were verified. Dataset contained 3,143 verified positive samples and 495 negative samples where no landing site was verified, rest of the 1,811 samples had no verification. We used those verified positive samples as our benchmark database and utilized 50% of those positive samples for training and the other half for model validation.

A-1 Google Static Maps API

We used Google static maps API for collecting imagery, which contains google earth imagery. The service is accessed by sending an HTTP request with a query containing the desired parameters, which are responded with an image based on the parameters. The parameters used are center, zoom, size, and map type. Center determines the coordinates that should be the center of the image. Zoom determines the distance a pixel will represent. Size determines the number of pixels in the image. Map type determines which type of image should be retrieved (as Google maps contains road maps). For the purposes of this project, size was always set to the maximum value of 640x640, and the map type was always satellite. Center was set to the desired coordinates to be sampled for the image and depending on the case can represent either a helipad location or a location where a helipad is not expected. The most detailed images are at a zoom of 20, however a zoom of 18 was used instead. The difference between the two zooms can be seen below in Figure 2. There is a cost associated with making API calls beyond a certain limit, so efficiency of calls will become important when scaling up.

Figure 2

Difference in Zoom Levels

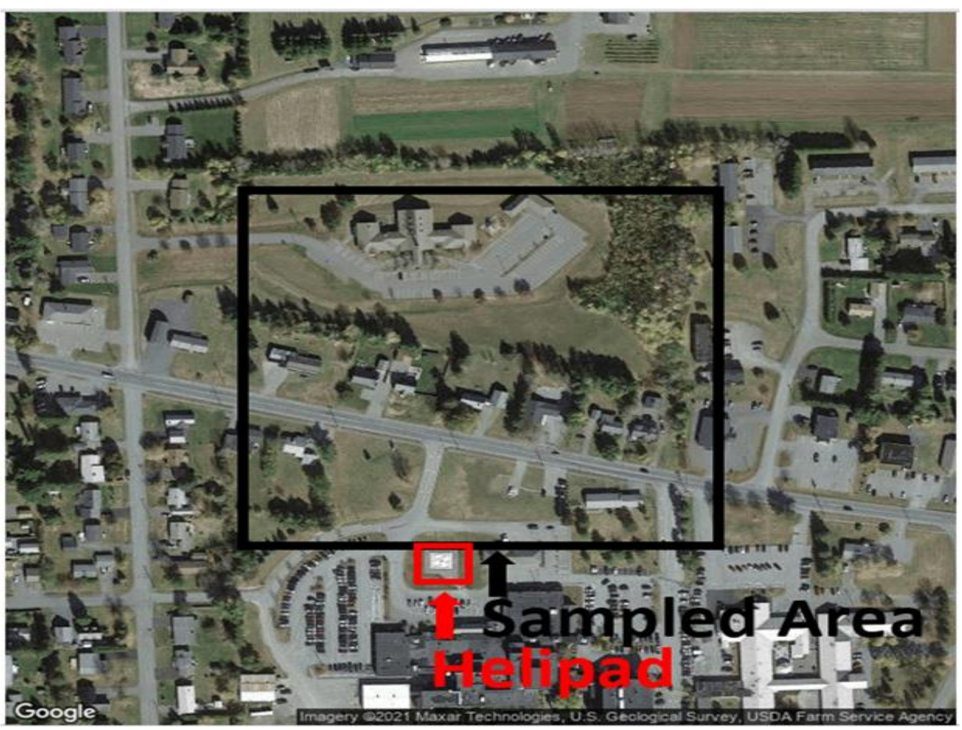
(1) : Zoom Level 18 (2) : Zoom Level 20

A-2 Issues Encountered During Data Acquisition

One of the issues we noticed was margin of error with positive samples, not all positive samples were centered and off by a factor but because not all helipad samples were centered that added a level of diversity to our dataset where some helipads were in different corners of the image. We also noticed some helipad locations being off by a bigger factor and being partly or completely outside of the collected image frame. Figure 3: shows a location that was nearby reported coordinates but off by a factor big enough that it was not visible in the 624 by 624 image.

Figure 3

Non-Usable Location with Helipad Nearby



Another issue with the service is collecting larger imagery. The maximum size of an image is limited to 640x640 pixels. This was the size used during the annotation process, and thus was the size collected. However, while this allows for larger errors in the coordinates, this does also cause an issue when trying to search for and localize helipad locations.

There is also an issue of recency. The images used in google maps are not real time images, but rather imagery taken during a survey. This means that the overhead view that was sampled does not actually reflect the current state of the area. Google attempts to keep the images up to date such that the available imagery should be less than 3 years old, however that is still a long period of time. This does limit the algorithm's effectiveness for determining the accuracy of new entries of recent helipads, as the available imagery may come from a point before the helipad was constructed.

Lastly not every location has the same image quality or zoom size, which leads to some images with poor quality and various zoom levels, this also limits effectiveness of training a model at a certain zoom level. Following are some examples of inconsistent satellite imagery.

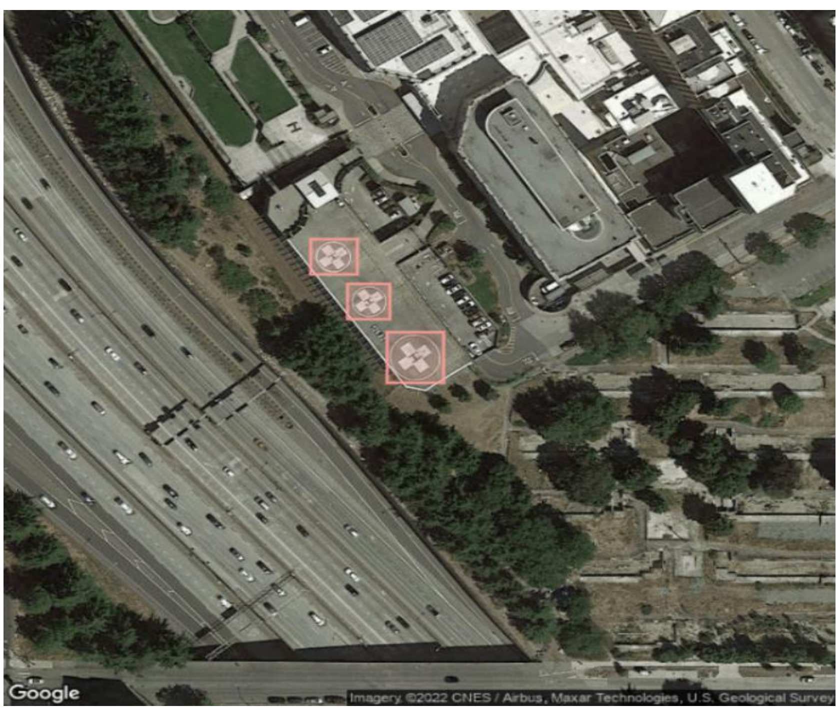
II-B Labeling

MakesenseAI is an online platform that assists in labeling data for object detection and image segmentation [insert reference]. MakesenseAI provides outputs in various formats that support different model frameworks, hence making it easier to train various models.

MakesenseAI was used to manually create bounding boxes around helipads. Bounding boxes simply indicate the area of interest for the model for each given object type. Each bounding box has its image coordinates and object type, this information allows the model to localize each object and its type with its respective image. An output file is created for each image containing rows for respective bounding boxes of that image, and the model associates this file with its respective image to retrieve this information. Figure 4 illustrates an image with a bounding box; multiple bounding boxes can be drawn on an image.

Figure 4

Labeled Helipads



Object detection models have different requirements for file formats of annotated label files such as VOC XML, CSV, or single CSV. MakesenseAI allowed us to export our annotations in required VOC XML and CSV format hence only having to label images once.

II-C Benchmark Dataset

After the above collection, labeling, and curation steps, a helipad identification benchmark dataset was created. The positive set contains 3,343 samples. FAA’s database consisted of different types of landing areas including helicopter parking pads, helidecks, EHLFs, and heliports. These sites included urban and rural landing locations as well as some maritime locations. Some helipad types such as hospital landing sites are more represented than others, but the dataset was very diverse for most part. *Figure 5* shows some of the images in the dataset.

Figure 5

Dataset Examples

Chapter 3

Helipad Detection from Satellite Imagery

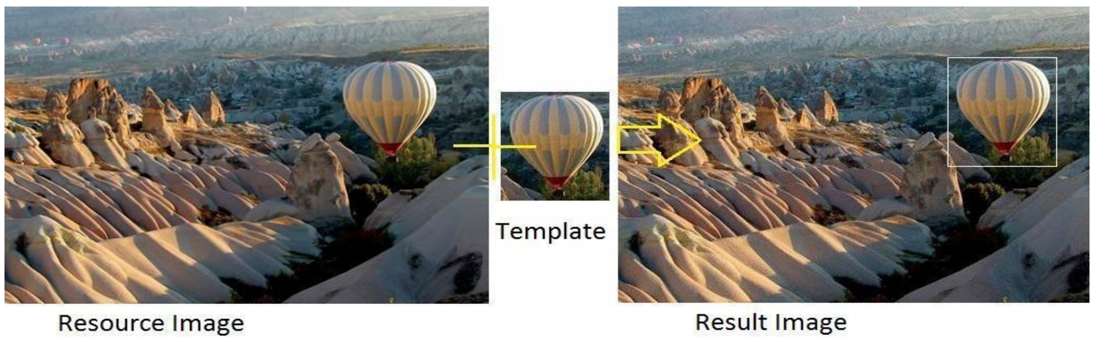
III-A Traditional Computer Vision Techniques

Digital Image Processing has long relied on using conventional computer vision techniques that have proven very successful in object detection problems such as documents scans, detecting license plates and parking occupancy detection. Computer vision techniques don't require having a dataset for training and can produce good results with just a few training examples, some of these computer vision techniques include SIFT, SURF and Template Matching. For our purposes we only explored Template Matching.

Template matching is used for finding parts of an image that matches a template image, in our case that would be a template helipad sample. It is a much simpler solution than a neural network as it does not require extensive manual data labeling for training, template matching does a pixel-by-pixel comparison of an image with a template matching. It creates a similarity matrix for each section of the image and if similarity of matrix reaches a certain threshold that section is identified as a template object. So, it is possible to achieve object detection using template matching especially if you are only dealing with one class, such as the helipad in our case. Figure 6 shows an example of this technique

Figure 6

Template Matching Example



A-1 Template Matching

There are several template matching techniques that can be implemented using python's OpenCV library, but we experimented with the following two correlation methods.

Figure 7

Correlation Methods

TM_CCOEFF_NORMED
Python: cv.TM_CCOEFF_NORMED

$$R(x,y)=\frac{\sum_{x',y'}(T'(x',y')\cdot I'(x+x',y+y'))}{\sqrt{\sum_{x',y'}T'(x',y')^2\cdot\sum_{x',y'}I'(x+x',y+y')^2}}$$

TM_CCORR_NORMED
Python: cv.TM_CCORR_NORMED

$$R(x,y)=\frac{\sum_{x',y'}(T(x',y')\cdot I(x+x',y+y'))}{\sqrt{\sum_{x',y'}T(x',y')^2\cdot\sum_{x',y'}I(x+x',y+y')^2}}$$

We implemented template matching by using a helipad template image, as a result we found a successful case where template matching was able to identify the helipad but also ran into some issues that restricted practical implementation of template matching. We will discuss why we ran into these issues and as a result decided to explore convolutional neural networks for a more appropriate solution.

A-2 Template Matching: Results

Figure 8

Template Matching Results #1

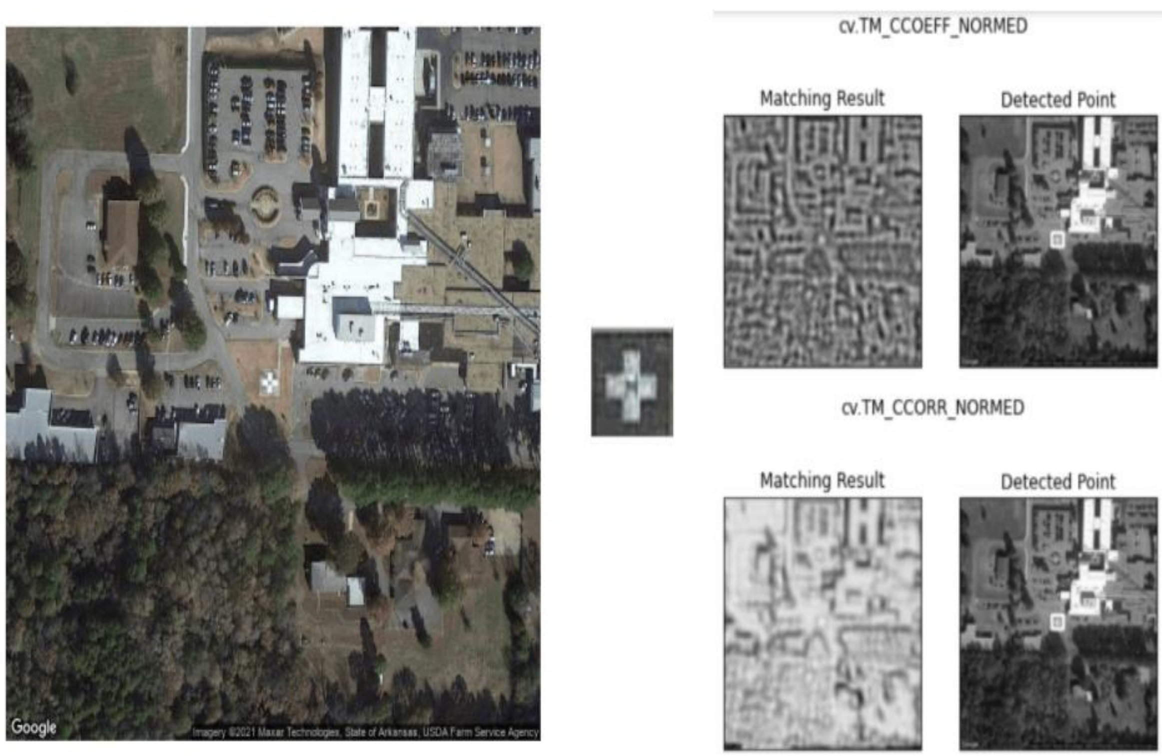
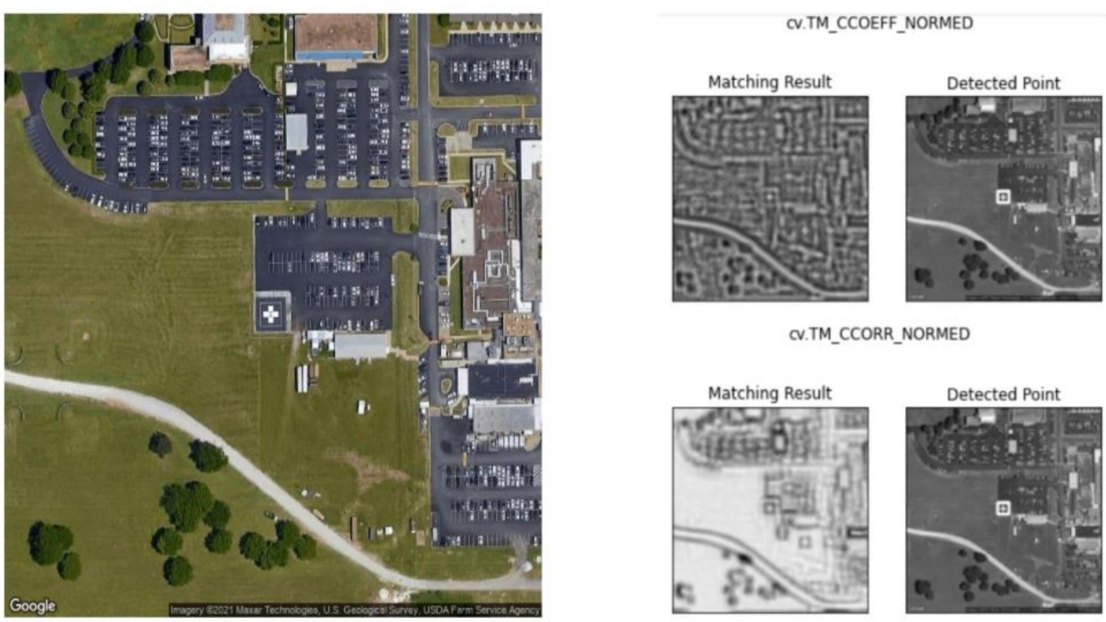


Figure 9

Template Matching Results #2

Figure 10

Template Matching Results #3



You can see that we used a template image that was not part of successful cases, but template matching was still able to identify those helipads due to high similarity with the template. But this approach fails when it encounters helipads with a slight variation or helipads with a rotated angle, which makes sense as the similarity between those images and template drops significantly. Figure 11 and Figure 12 show cases where template matching fails to locate helipads with different rotation angles despite having a similar template image.

Figure 11

Template Matching Results #4

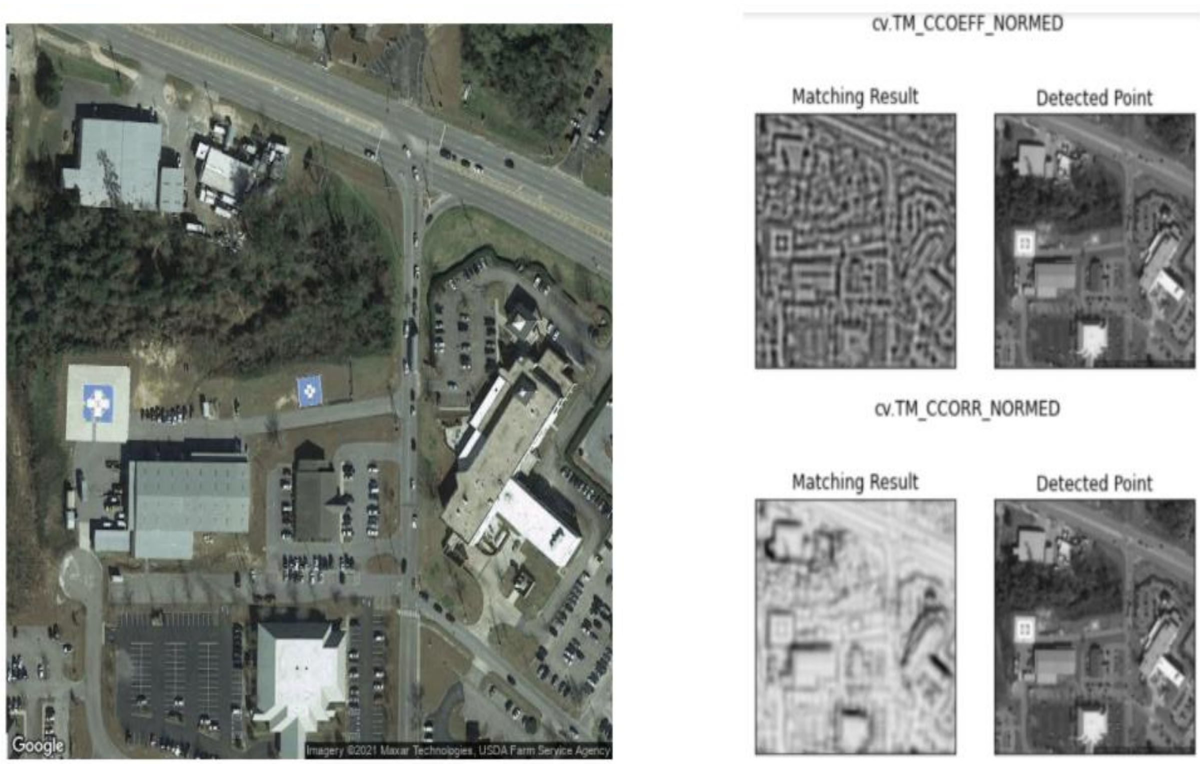
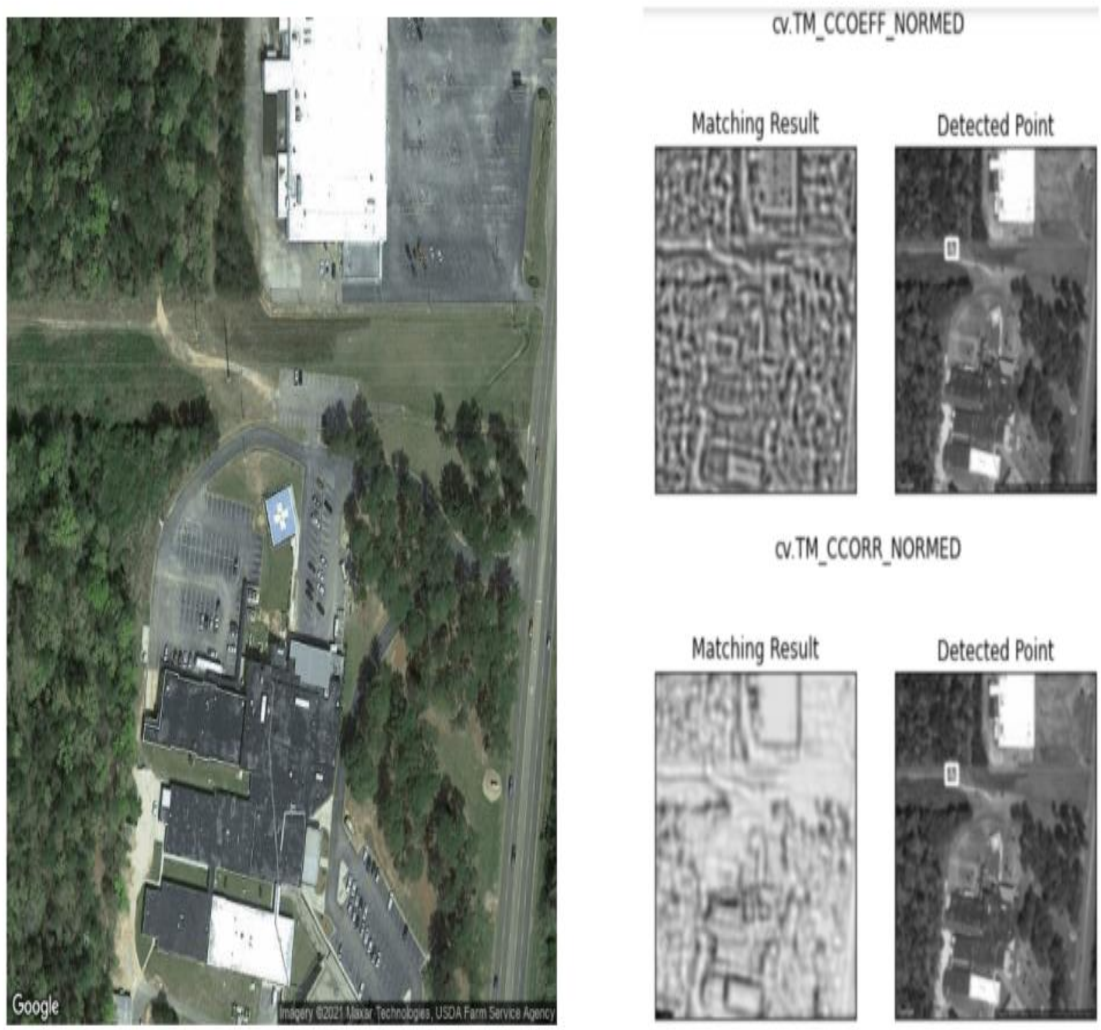


Figure 12

Template Matching Results #5



We also observed the same issue with a different template image of a circular helipad where template matching fails when helipads are at rotation angle despite being circular in pattern.

Figure 13

Template Matching Results #6

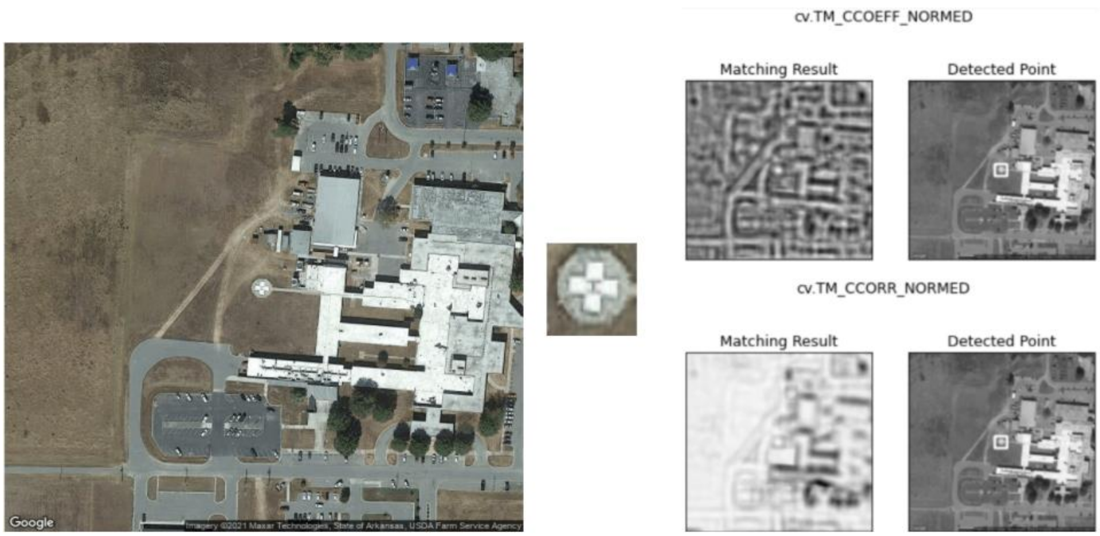


Figure 14

Template Matching Results #7

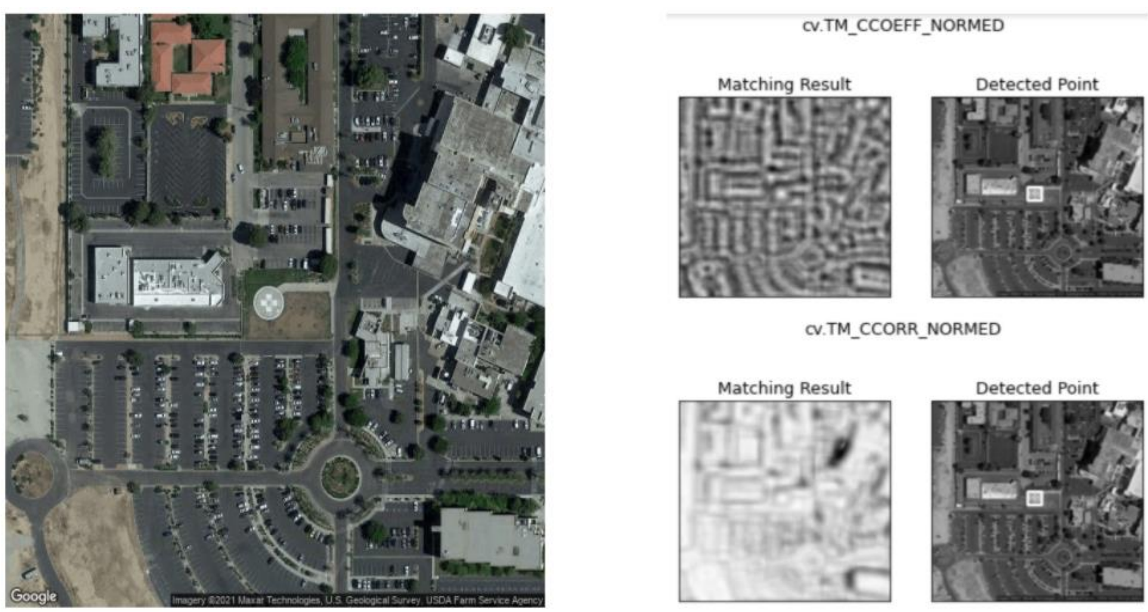


Figure 15

Template Matching Issues #1

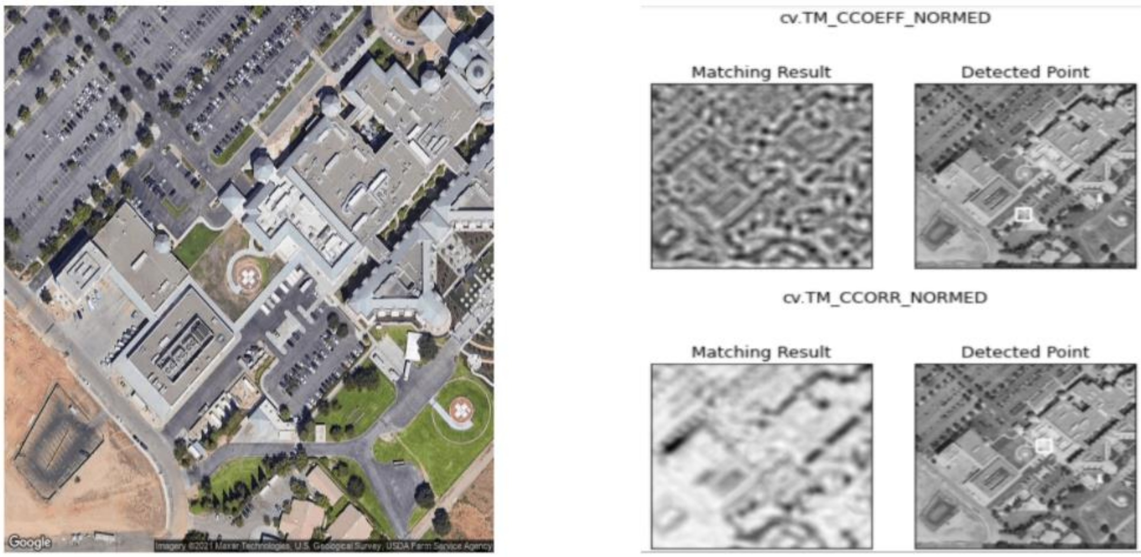
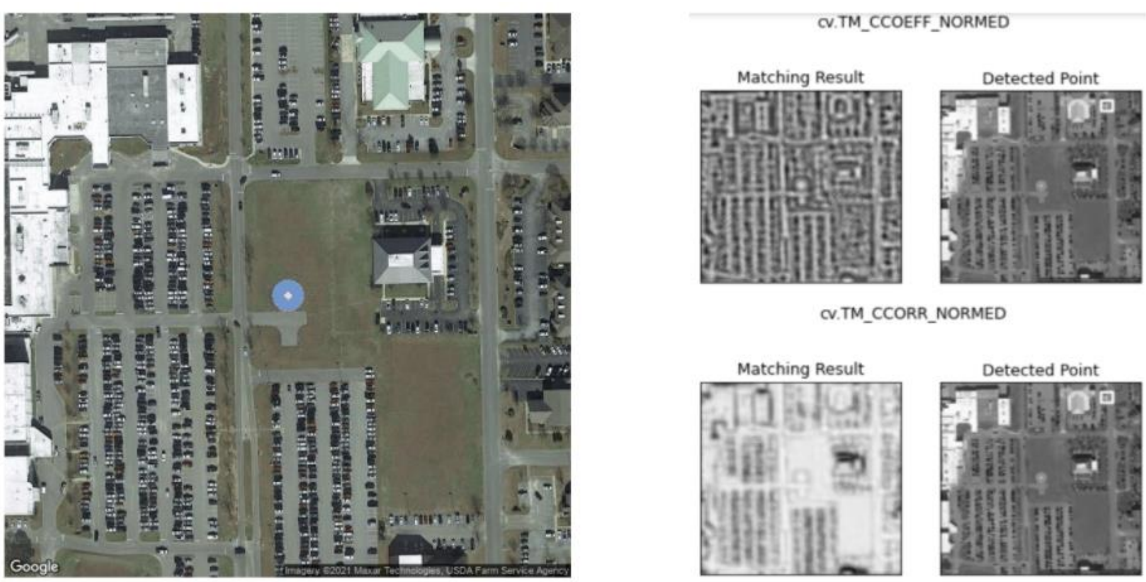


Figure 16

Template Matching Issues #2



Despite showing some promising results template matching fails whenever there is slight deviation from template and you need to add a template for each type of helipad pattern, which includes having templates with all possible rotation angles. This requires extensive manual verification of benchmark dataset to make sure all different templates are included, but despite this there are satellite imagery inconsistencies that result in helipad patterns deviating from any set of helipad templates.

Hence, we decided to explore conventional neural networks and use deep learning to find a more appropriate solution for our requirements. We will now discuss “Deep Learning” and how we used satellite imagery to train and validate our models.

III-B Deep Learning

Deep Learning is a subset of Artificial Intelligence based on artificial neural networks; deep learning excels in many areas including representational learning. Deep learning consists of neural networks with three or more layers, it leverages a training dataset where a model aims to “learn” over iterations. Compared to traditional computer vision techniques, deep learning provides optimal solutions for tasks such as image classification, semantic segmentation, and object detection.

There are many objects detection models that have proven their efficiency in recent years with different tradeoffs and advantages. We wanted to test and compare different models and chose a model that was optimal for our requirements. We decided to train three different models on a small dataset of 200 images, these images were manually labeled, and their truth values were known. We trained Detection’s Faster R-CNN, TensorFlow’s SSD and YOLO V3, we now discuss these models and our comparison results.

Faster R-CNN is a very popular convolution architecture and has shown excellent results in object segmentation and detection problems. Faster R-CNN uses a region proposal network (RPN) for generating region proposals and a network using these proposals to detect objects. The time cost of generating region proposals is much smaller in RPN compared to other models. RPN ranks region boxes and proposes the ones most likely containing objects.

TF- SSD architecture is a single convolution network that learns to detect objects in one pass. The SSD network consists of base architecture (MobileNet in this case) followed by several convolution layers. SSD architecture allows faster processing compared to other models with a tradeoff for accuracy.

YOLO model architecture is one of the most popular models used for object detection and is known for its high accuracy and fast computation speeds. This model is based on the idea that a single network predicts bounding boxes and class probabilities directly from full images in one pass. This allows for end-to-end optimization specifically on detection performance.

Figure 17

YOLO V3 Network Architecture

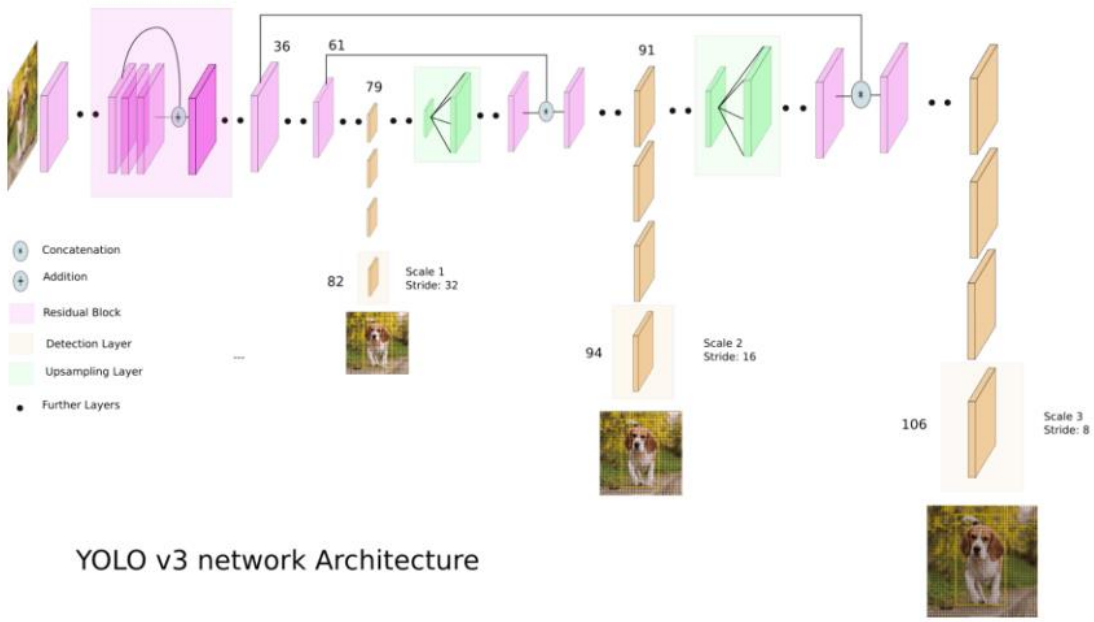


Figure 17 displays the YOLO v3 model architecture. The input is a batch of imagery data, in our case it will be places of interest with the potential to contain helipads. The output is a list of bounding boxes along with recognized classes. Then utilizing intersection over union, or the overlap of the predicted bounding box compared to the ground truth labels, a confidence score can be given on the accuracy of the detection. The model is 53 layers deep and pretrained on the ImageNet model.

Before we discuss our results and comparison on these models it's important to understand a few accuracy metrics, metrics used here are True Positive, False Positive and False Negative. These metrics are intuitive, but we also added another metric that we called "Unseen False Negative".

Figure 18 shows an example of a true positive as the predicted bounding box overlaps the ground truth, for our purposes we set a threshold of 33% overlap between predictive and ground truth bounding box to consider a detection true positive.

Figure 18

True Positive

Figure 19 shows an example of false positives as the model falsely identifies a helipad area.

Figure 19

False Positive

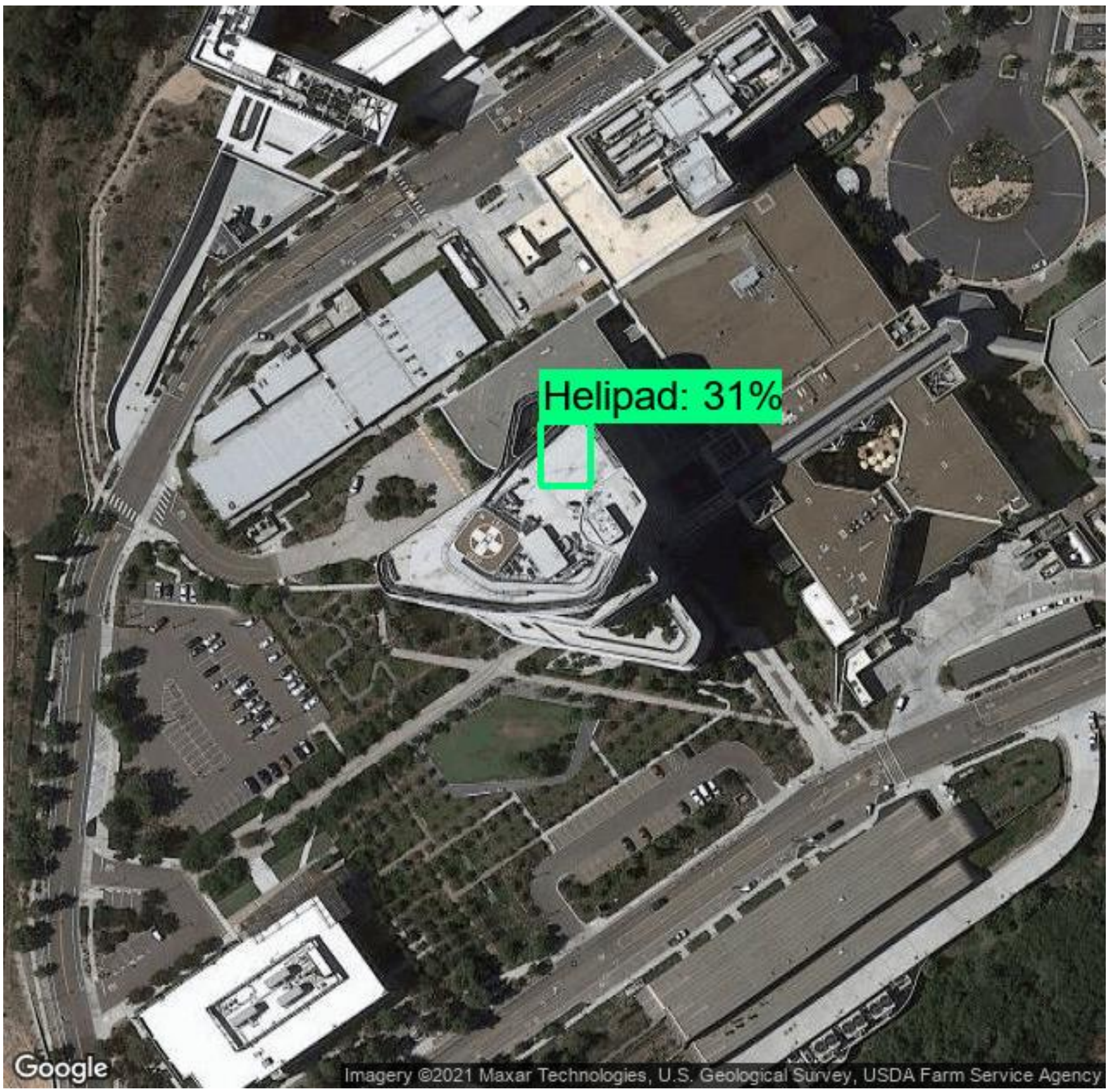


Figure 20 shows an example of False Negative where the model identifies an image with no helipads and misses a true positive.

Figure 20

False Negative

In our validation data we also included positive samples that models had not been trained on and were slightly different from the training set, this gave us an additional insight to different model performances. Figure 21 shows an example of this, where the image contains helipad patterns that were not part of the training sample.

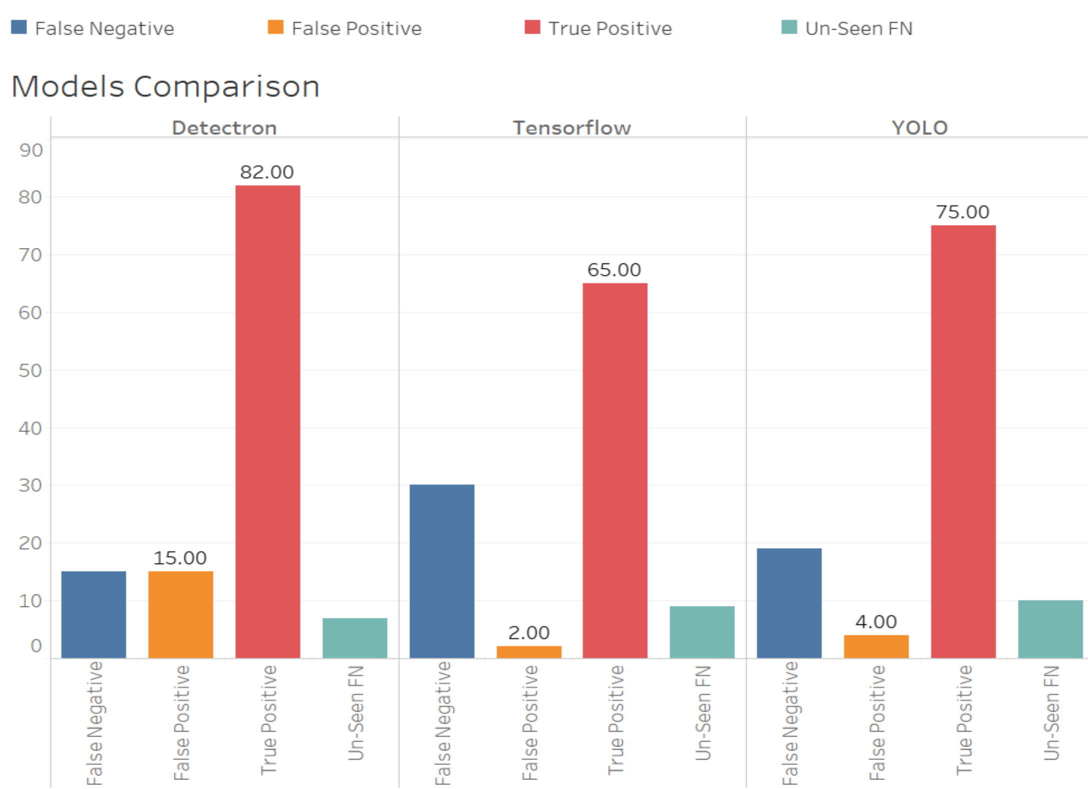
Figure 21

Unseen Images

Figure 22 shows comparison results of three models, detectron' Faster RCNN detected the greatest number of positive samples but also had the highest false positive count. TensorFlow's SSD had the lowest false positive count but also a low true positive count and high false negative count. YOLO V3 had a decent true positive and false negative ratio, as well as low false positive count. All models performed relatively well with "unseen" images and had low unseen false negatives, which shows the ability of deep learning to generalize a problem to an extent and handle small variations in testing dataset. Based on these metrics we decided to move forward with YOLO V3 as it had promising results with low false positives and high true positives.

Figure 22

Models Comparison



III-C Detection Results

We manually annotated the rest of our benchmark dataset and set up a training dataset of about 1,771 images which contained 200 LZControl and 1,571 FAA 5010 database images. We manually vetted this dataset to remove some inconsistencies and discrepancies, we trained YOLO V3 on this dataset and used the rest of our benchmark database as validation set.

We thoroughly evaluated our model's performance on validation dataset by calculating precision and recall. Figure 23 shows how these metrics are calculated

Figure 23

Accuracy Metrics

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Figure 24

Accuracy Results

	PRECISION	Recall
YOLO V3	76.56%	74.95%

YOLO V3 showed satisfactory numbers for precision and recall. Since all the validation dataset images contained positive samples, we can narrow down the model's performance to true positives and false negatives. Where true positives are images that were successfully identified as positive samples and false negatives where our model failed to identify positive samples. Model had 85% true positives and 15% false negatives accuracy on validation dataset.

Figure 25

Accuracy on Validation Dataset

Data	Manual Verification	Model Detection
FAA 5010	1,572	1,342

C-1 True Positives Detected by Model

Following are some examples of true positives detected by model, these include different patterns, images with multiple helipads and some images that were slightly different than training set. Figure 29 shows a good example of model detecting new patterns based on similarity with training set.

Figure 26

True Positives Detected by Model #1

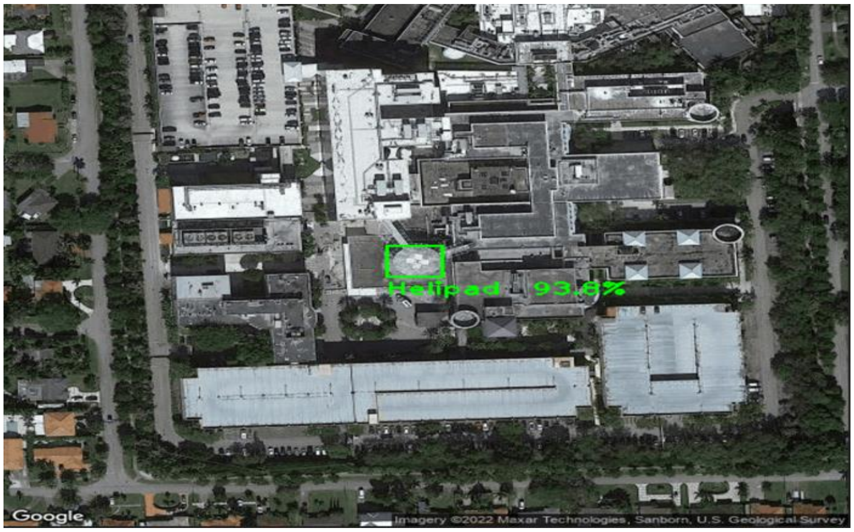


Figure 27

True Positives Detected by Model #2

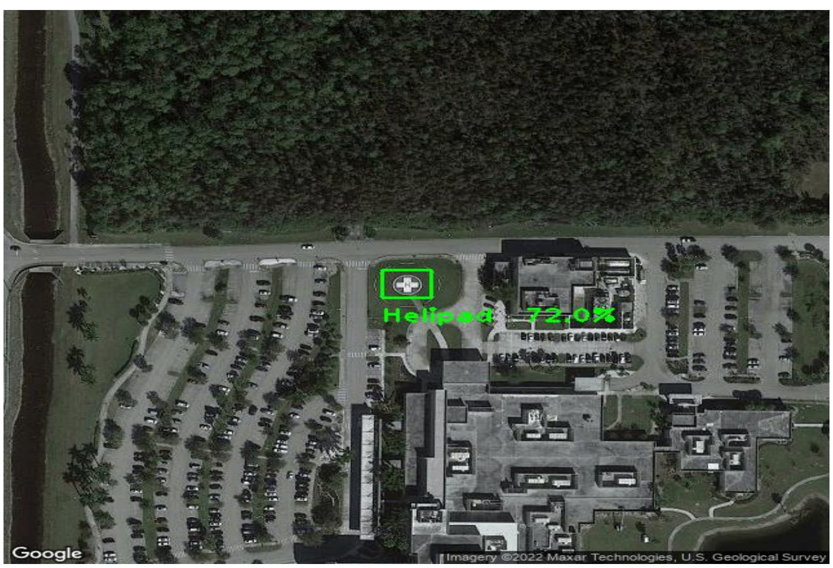


Figure 28

True Positives Detected by Model #3

Figure 29

True Positives Detected by Model #4

Figure 30

True Positives Detected by Model #5

Figure 31

True Positives Detected by Model #6

As shown in above images our model detected different types of helipads and performed well in detecting helipads in different terrains as well. Some images contained multiple helipads but for our accuracy metrics we counted each image as one positive sample.

We also analyzed false negatives to understand why the model missed those samples, these false negatives could be helpful in retraining the model in future works. We noticed a few issues in false negatives, these issues included poor imagery resolution, un-trained helipad patterns, and false zoom level imagery. These issues limit the model's performance as Google Maps' imagery is not consistent with all areas. Images that contained new helipad patterns that weren't part of the training cycle can be used as feedback for future works. Following are some examples of false negatives

C-2 API Issues

Figure 32

API Issues #1

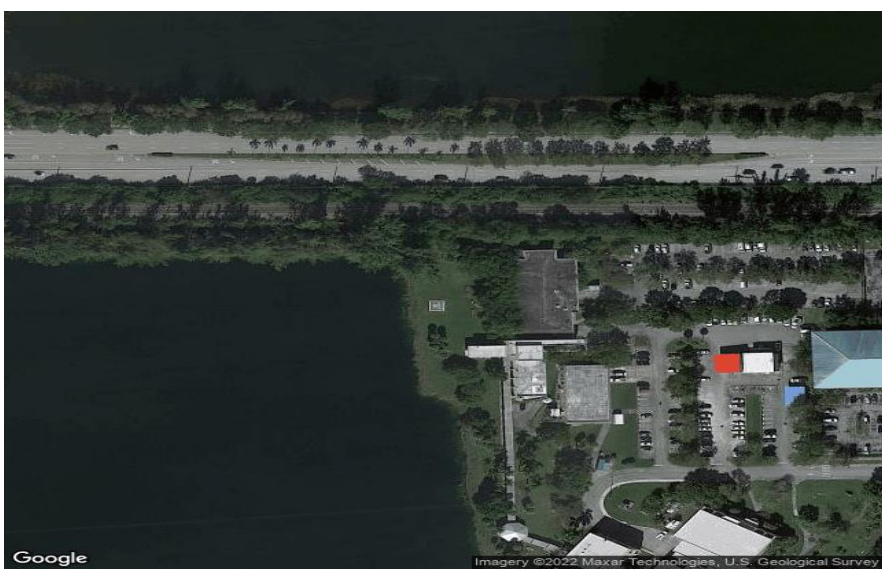
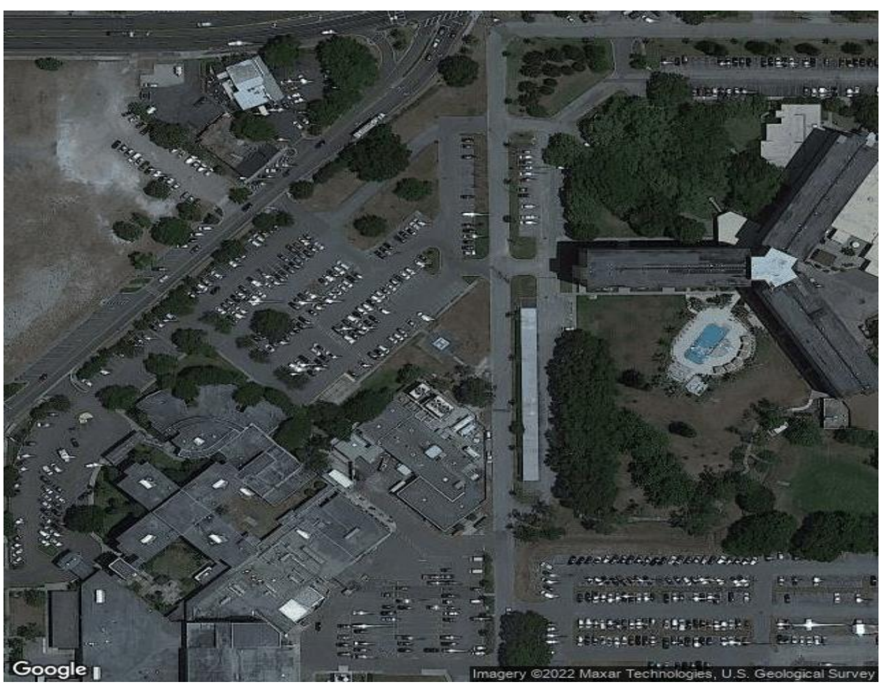


Figure 33

API Issues #2

Figure 34

API Issues #3



C-3 Untrained Helipad Patterns

Figure 35

Untrained Helipad Patterns #1

Figure 36

Untrained Helipad Patterns #2

Figure 37

Untrained Helipad Patterns #3

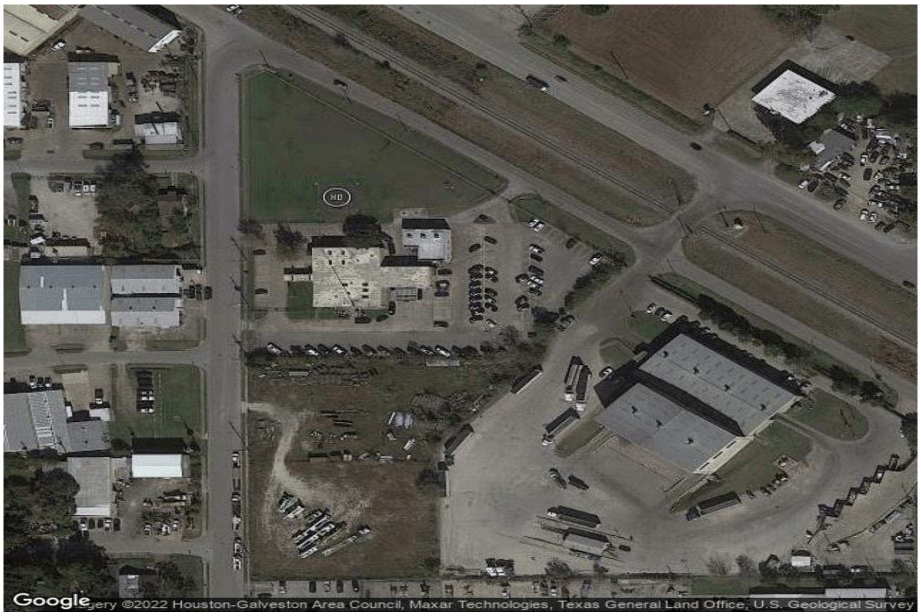


Figure 38

Untrained Helipad Patterns #4

In conclusion false images consisted of images that were poor resolution, zoom level and untrained helipad patterns. With satisfactory results on the validation dataset, we were ready to deploy our model and develop tools to meet FAA requirements. We will discuss our applications of this model which were focused on scaling the FAA's current database by searching for new helipads.

Chapter 4

Search For New Helipads

The algorithm developed was reliable enough for us to move forward with it and develop tools that could utilize it, with the primary aim of scaling the FAA's current database of helipads. We wanted to have the ability to scan any given area and search for helipads in that vicinity. Our approach was to input a set of coordinates and then collect satellite imagery with a certain mile radius and run model inference on that imagery to detect helipads. This would help us find missing helipads in the 5010-database due to reporting errors or coordinates being off by an error factor. We specifically wanted to apply this approach to completely verify the FAA's 5010 database which contained 495 negative samples and 1,811 samples with no verification. We could use this tool to scan these sites and search for a helipad in a certain radius vicinity.

We needed the ability to scan bigger areas around a set of coordinates to run inference on that imagery, one possible way to do this would be to use a smaller zoom level so imagery would cover a bigger area. However, this approach would not work as the model was trained on zoom level 18 and would fail to perform if run on much smaller zoom levels. Our second option was to apply a “Collage” approach, an approach developed in previous works of this project. We will not discuss the details of this approach development and refer to our previous work for further explanation [2]. This approach samples a region numerous times around a given set of coordinates and combines this information.

While this approach is not very inefficient as it requires numerous singular API requests for each site and is time consuming, it does maintain the integrity of the zoom level that our model requires. Future works could revisit this approach and develop a more efficient way of collecting bigger area imagery.

IV-A Search and Scan

Using the collage approach, we were able to collect imagery around a set of coordinates by setting any desired mile radius area. After collecting desired imagery, we used our model to scan and search for helipads in collected imagery. Our algorithm would output identified images with helipads along with their centered coordinates, all these output images could be then manually verified to confirm positive samples. Once validated we could deliver these newly found sets of coordinates to the FAA. This would enable us to search for helipads whose coordinates were off in the database and could now be corrected. Following images demonstrate an example of this approach.

Figure 39

Output Image with Set of Coordinates

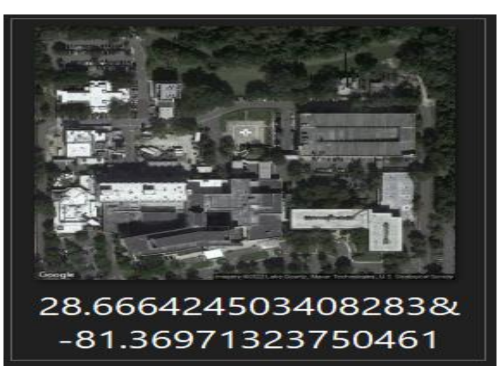
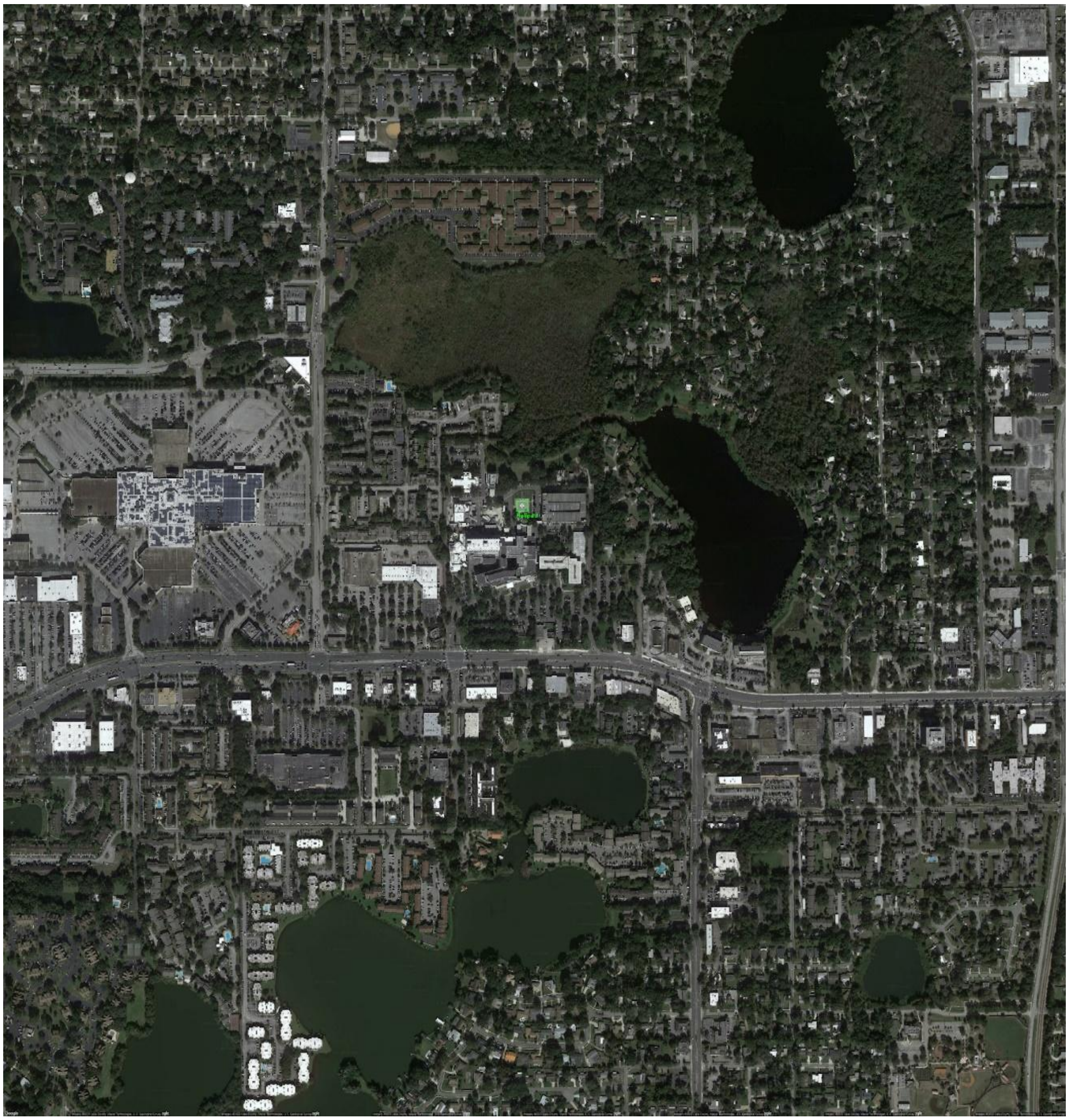


Figure 40

3 Mile Search Area



Once set up we applied this algorithm on the FAA's 5010 database, we scanned and searched 495 negative labeled and 1,811 unknown sites with a search radius of 3 miles. This was a good “real world” test for our model as these coordinates consisted of diverse terrain and regions. After manual verification and cleanup of collected potential positive samples, we attained following results.

Figure 41

Scan and Search Results

5010 Dataset	Helipads Detected
Unknown – (1811)	400
Negatives – (495)	62

We successfully corrected and added 462 helipad landing sites to the FAA's current database, all these sites were manually verified and delivered to the FAA.

During this approach we observed some patterns in false positives that appeared over and over, these false positives could be used to train models on true negatives in future works. These patterns included soccer fields, golf courts and water towers. Following are examples of few false positives detected.

Figure 42

False Positives #1

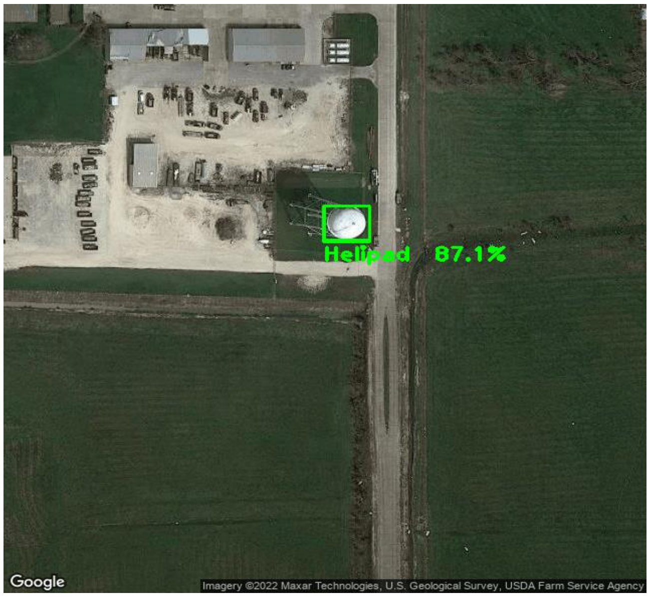


Figure 43

False Positives #2

Figure 44

False Positives #3

Figure 45

False Positives #4

Figure 46

False Positives #5

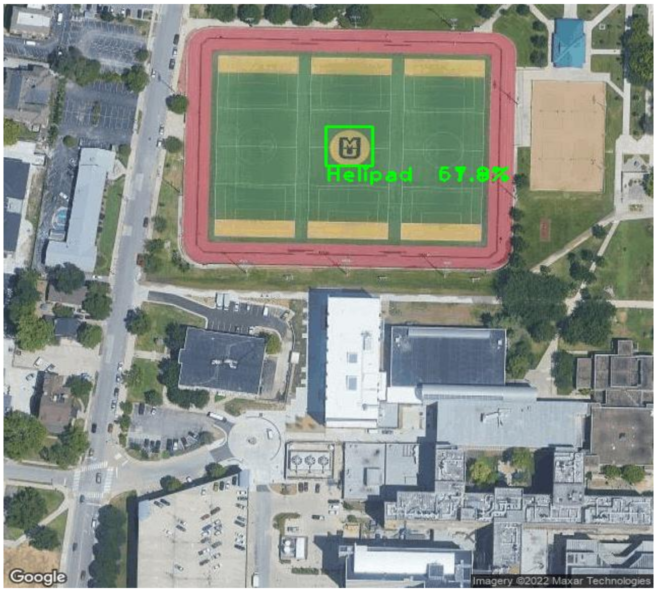


Figure 47

False Positives #6

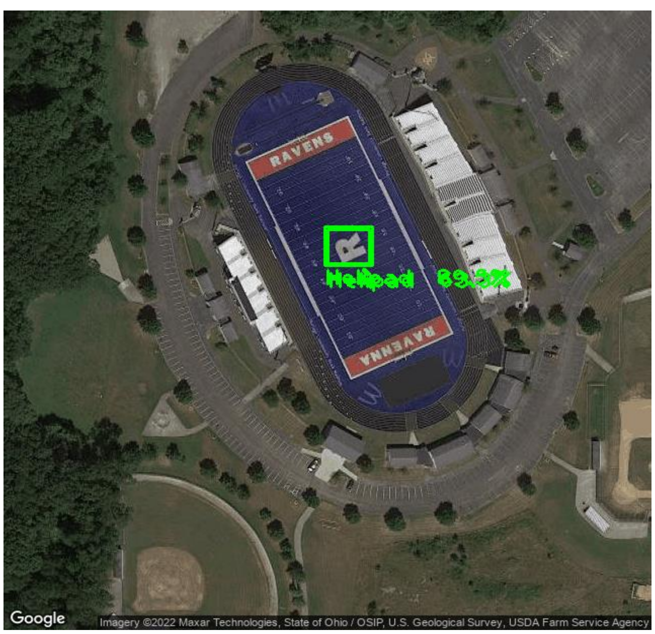


Figure 48

False Positives #7

Figure 49

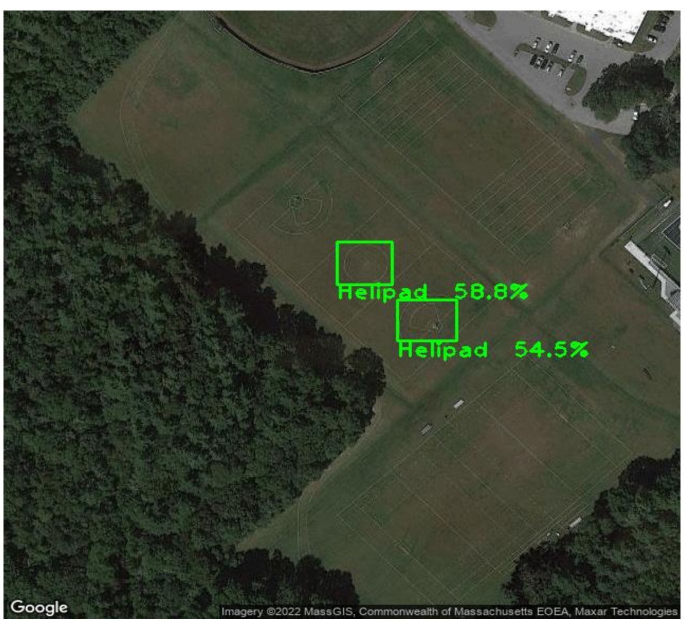
False Positives #8

Figure 50

False Positives #9

Figure 51

False Positives #10



IV-B Other Implementations of Our Model

The FAA believes there are over 2,000 missing hospital helipads in the US, using Google's API we retrieved coordinates of over 800 US cities, we then pulled satellite imagery of all these sites. This dataset contained 9K images collected via Google Maps API, these images were collected at zoom 18. We used our model to filter this dataset and identify images with helipads. Our model filtered out 1,748 images as positive samples, we then manually annotated these images to get an accurate assessment of our model. We found 1,458 images containing one or more helipad landing sites, hence our model had accuracy of 83% true positives and 17% false positives. After removing duplicates, we delivered the FAA with a list of 972 newly detected hospital helipads, marking great success of our deep learning approach.

Our model can be used to further scale the FAA's database by collecting more imagery on areas of potential landing sites and filtering out potential helipad sites. Model's performance can be improved continuously by incorporating feedback and re-training our model.

Chapter 5

Conclusion and Future Works

V-A Conclusion

Aim of this project was to develop an algorithm that had the ability to identify helipads in satellite imagery, our secondary objective was to use the model and develop a process to find new satellite imagery. We achieved satisfactory results on both goals.

Our comparison of different object detection models helped us choose the appropriate model for our requirements, YOLO V3 produced fruitful results in identification of helipads using satellite imagery. We utilized FAA's 5010 database as our primary training and validation dataset, our model was trained on a diverse enough dataset that it produced satisfactory performance. We constantly evaluated and refined our dataset by manual verification throughout our training process to ensure a clean training dataset, it is crucial to minimize noise and bad samples in the training set. Our model demonstrated its efficiency by identifying additional hospital helipads from a large set of hospitals satellite imagery, manual inference of such a huge dataset would have been a very tall task.

Our model also performed well on our secondary objective of searching for new helipads given a region coordinate, we used our "collage approach" to collect imagery in each mile radius and scanned for new helipads. This approach also proved fruitful as we were able to find helipads that had incorrect coordinates or simply missing from the FAA's database, this was a critical requirement for this project. Our approach validated our model being used as an inference tool and opened the door for further search and identification of helipads in other areas of interest.

We have several suggestions for future works that could improve accuracy and efficiency of this system, this includes model training approach, improving training dataset and more optimal programming tools to utilize models for imagery inference.

V-B Future Work

We only explored 3 distinct models for object detection and trained them on a very brief dataset for performance comparison, we suggest future works thoroughly explore more models and use a bigger dataset for training and performance comparison. We also recommend comparing models based on not only their accuracy but also on computational and processing efficiency.

Future works should also improve training dataset by incorporating our feedback of false positives which shows clear patterns that can compromise performance, these patterns include, water towers, golf courses, soccer fields and solar panels. These images could be included in the training dataset as true negatives, hence reducing the chance of the model identifying them as positive samples in future works. We recommend further study of academic studies on using true negatives in object detection model training and reducing false positives.

Though our search and scan approach were successful, it can be improved by developing more cost-efficient tools as currently it relies on making single API requests to collect imagery of a region, a program with parallel requests and inference could drastically increase speed. We also recommend that future works consider the inference speed of different models as a performance metric. Improving the speed of this system would be critical if the model is to be used at a much bigger dataset or scan larger regions for searching new helipads.

Our work can also be used for FAA's other identification and localization problems such as detecting airports, identifying, and classifying different distinct helipad types and detecting objects landing hazards near a helipad.

References

[1] 150/5390-2c - heliport design, FAA, 2012.

[2]Specht, David, "Helipad detection from satellite imagery using convolutional neural networks" (2021). *Theses and Dissertations*. 2927.
https://rdw.rowan.edu/etd/2927