MASTER THESIS

In Order to Obtain the

PROFESSIONAL MASTER

In

GEOGRAPHIC INFORMATION SYSTEMS AND DATA SCIENCE

Presented and defended by:

Alihadi Hussein Zeineddeen

On Wednesday October 28, 2020

Title

OBJECT DETECTION AND MAPPING USING DRONES (CLUSTER BOMBS)

Supervisor
Ms. Gretta Kelzi

Reviewers
Dr. May Dhayni
Ms. Manal Sayed

Acknowledgments

I thank all those who in one way or another contributed in the completion of this project. First, I give thanks to God for protection and ability to do this work.

The internship opportunity I had with Esri Lebanon and Khatib & Alami Company was a great chance for learning and professional development. Therefore, I consider myself as a very lucky individual as I was provided with an opportunity to be a part of it. Special thanks to my Prof. Kifah Tout for giving me this opportunity and for his support throughout the year.

I would like to express my special thanks and gratitude to my supervisor Mr. Gretta Kelzi, and to Mr. Mazen Mrad and Mr. Hassan Mahmoud for all their support and guidance in completing my project. They were on my side whenever I faced new challenges during this work and their experience in many fields were always a great value.

I would also like to extend my gratitude to The Lebanese Army officers that were so cooperative and helped us in gathering the information and datasets we needed; And to my friend Kassem Shehade that helped me overcome many technical challenges.

And of course, my deepest thanks to my parents and my fiance Rayan that were my number one supporters.

Abstract

West Asia is one of the hottest zones in the world, where wars affect the daily lives of civilians even years after their end. In Lebanon the Lebanese army has been working for over 20 years to remove the unexploded cluster bombs left by the Israeli army, and over 2 years to remove the remains of the combats with the takfiri groups on the Lebanese-Syrian borders.
Here we report the exploitation of 2 worldwide-trending technical domains in the demining process held by the Lebanese army, which are the drone domain and Object detection (sub-field of AI) domain in a GIS positioning environment.
Our project includes gathering bomb images using different methods, training an object detection model, then taking a set of drone images covering a targeted demining area containing 'AO 2.5 RT' bombs in order to detect them and build a map including a feature class of the bombs' positions.

Keywords: Drones; Object detection; Machine learning; Keras-retinanet; Drone to map; GPS; bombs; demining; Map; feature class.

Table of Contents

List of Figures

Introduction

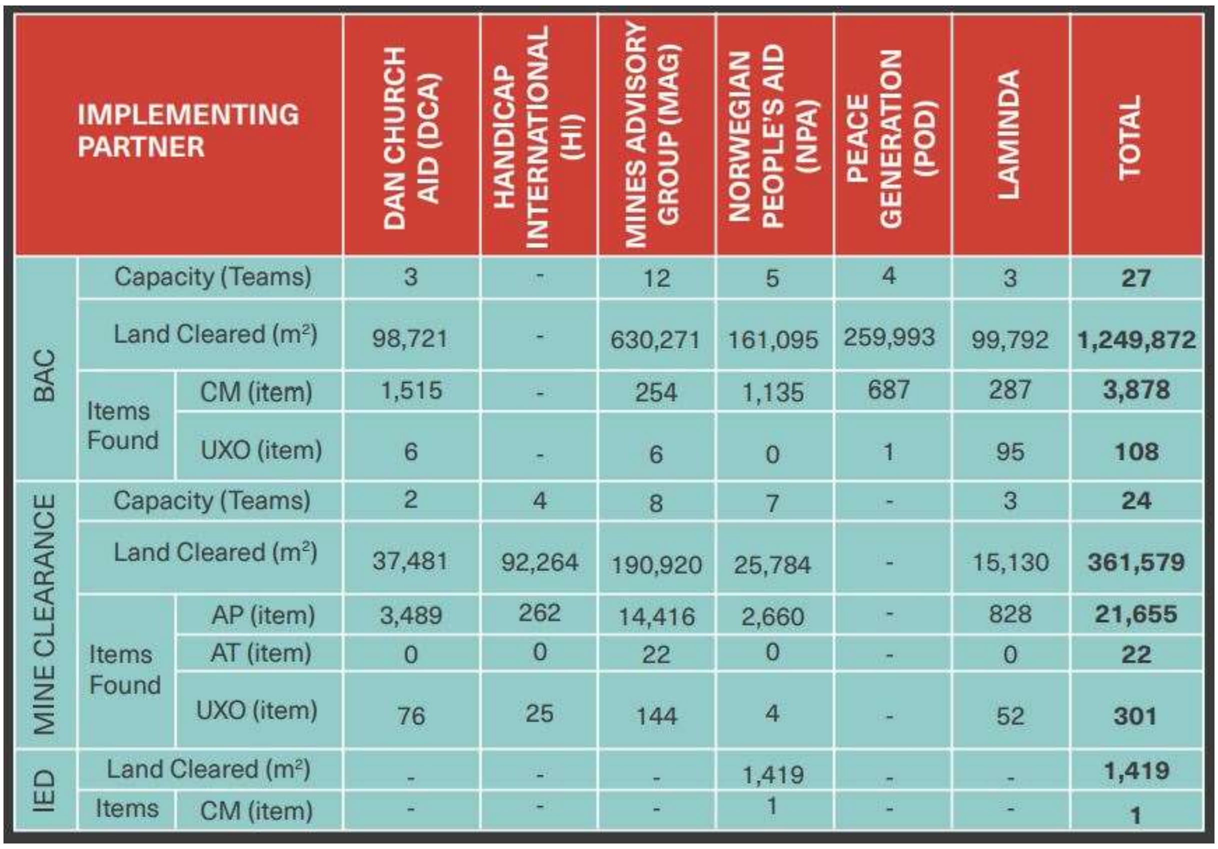
1. Background

1.1. Cluster Bombs in Lebanon

West Asia is one of the hottest zones in the world, where wars affect the daily lives of civilians even years after they end. In Lebanon the Lebanese army has been working for over 20 years to remove the unexploded munitions left by the Israeli army, and over 2 years to remove the remains of the combats with the takfiri groups on the Lebanese-Syrian borders.

1.1.1. Statistics

At the beginning of 2019, the total CHA (Contaminated Hazardous Area) was **46.217 million m2**. The teams of LMAC were able to clear more than 1.5 M m2 by the same year as shown in the figure below.



IMPLEMENTING PARTNER			DAN CHURCH AID (DCA)	HANDICAP INTERNATIONAL (HI)	MINES ADVISORY GROUP (MAG)	NORWEGIAN PEOPLE'S AID (NPA)	PEACE GENERATION (POD)	LAMINDA	TOTAL
BAC	Capacity (Teams)		3	-	12	5	4	3	27
	Land Cleared (m²)		98,721	-	630,271	161,095	259,993	99,792	1,249,872
	Items Found	CM (item)	1,515	-	254	1,135	687	287	3,878
		UXO (item)	6	-	6	0	1	95	108
MINE CLEARANCE	Capacity (Teams)		2	4	8	7	-	3	24
	Land Cleared (m²)		37,481	92,264	190,920	25,784	-	15,130	361,579
	Items Found	AP (item)	3,489	262	14,416	2,660	-	828	21,655
		AT (item)	0	0	22	0	-	0	22
		UXO (item)	76	25	144	4	-	52	301
IED	Land Cleared (m²)		-	-	-	1,419	-	-	1,419
	Items	CM (item)	-	-	-	1	-	-	1

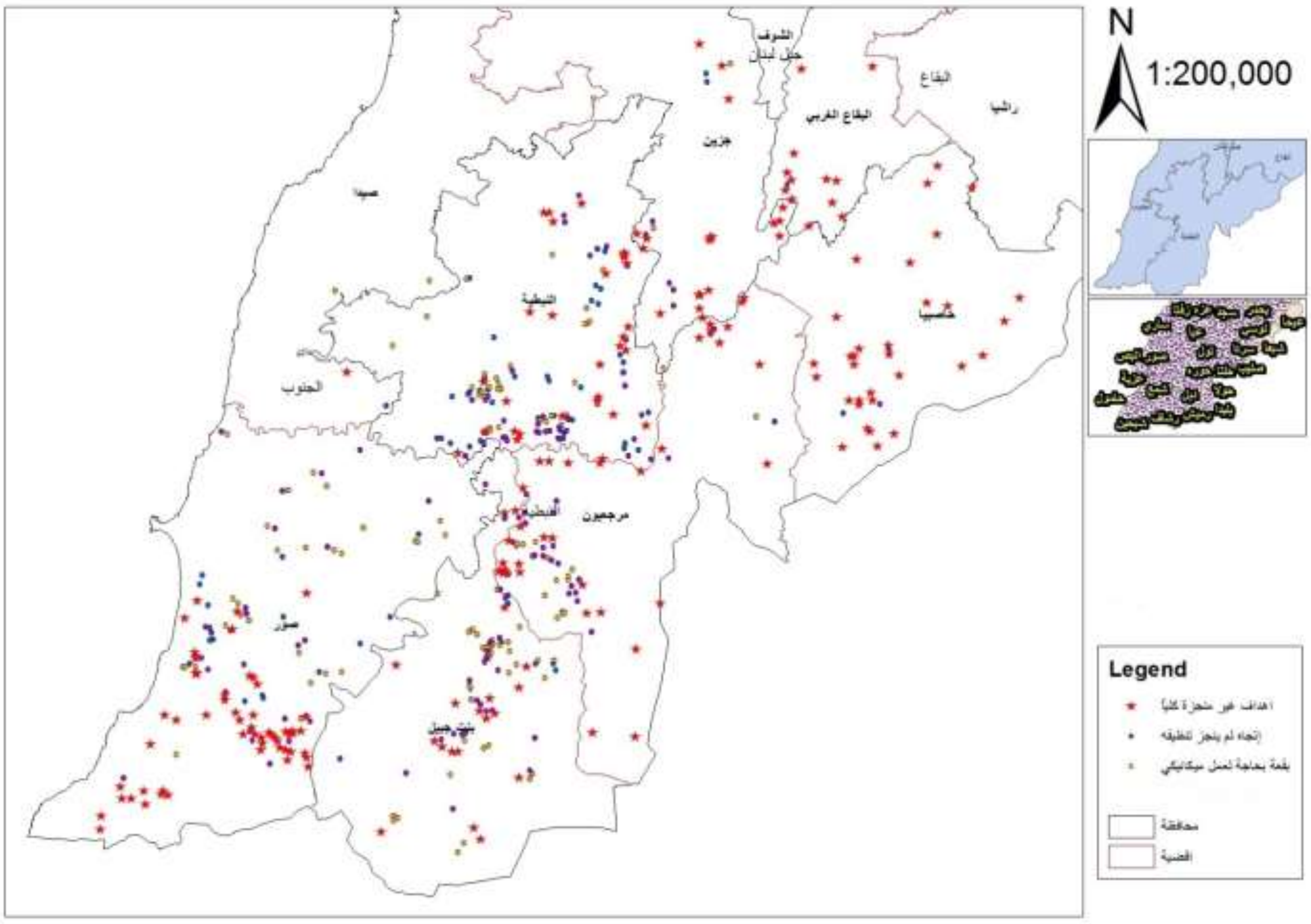
In addition to the previous statistics, a total of 132558 m2 were cleared by the Lebanese Army, divided into:

- minefields : 121398 m2
- cluster bombs: 11160 m2

By the end of 2019 LMAC has had 21,708 Anti-Personnel Mines removed. However

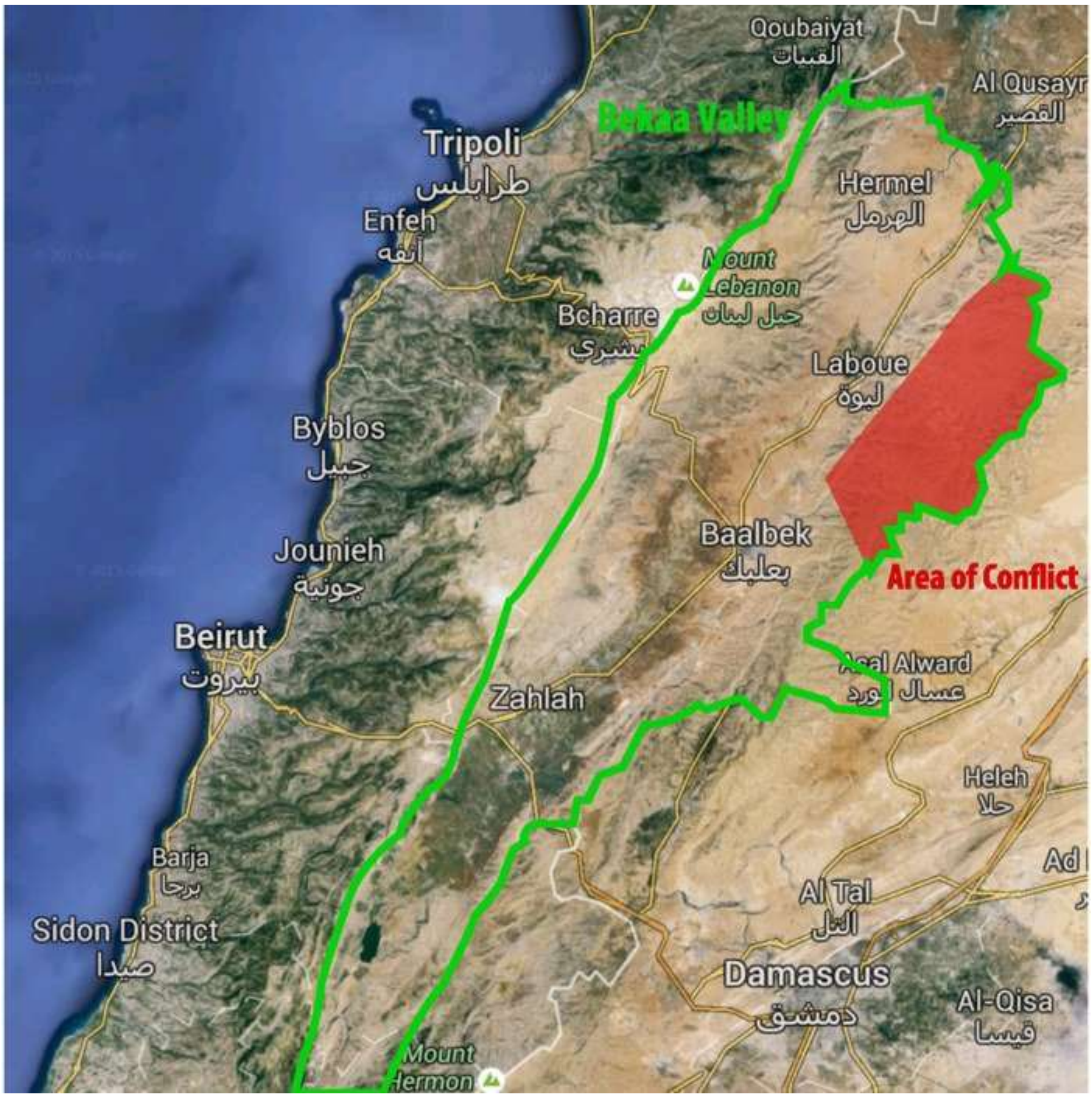
1.1.2. Geographic Distribution Contaminated Areas

Cluster Bombs fields in Lebanon are mainly found in South Lebanon and at the Lebanese-Syrian borders.



The map above shows the CHAs in South Lebanon that are still contaminated since the 2006 Israeli attacks on Lebanon.

Beside the areas in the south, the Lebanese Army Forces have managed to liberate a total of 120 Kilometers occupied by Daesh in Beqaa after Fajr el-Joroud operation that took place in 2017. The liberated land shown in the next map is being checked and cleared from all munitions by LMAC.

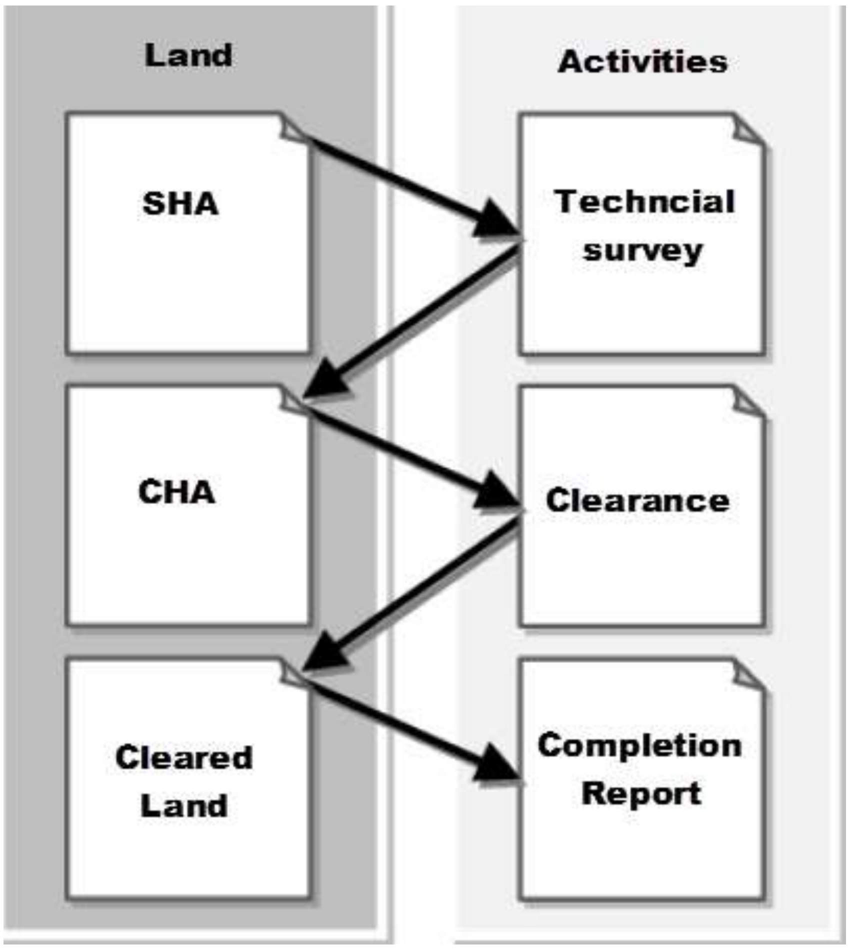


1.2. Demining Workflow

The clearance of contaminated areas has to be an organized and well-planned process for several reasons like:

- Avoiding human injuries during the clearance
- Efficient exploitation of limited resources
- Accurate (guaranteed) clearance results

This is an example for a standard workflow used by the demining teams to enhance their clearance operations.



The workflow consists of 6 elements:

1- **SHA** : Suspected Hazardous Area, which is an area that may contain unexploded munitions

2**- Technical survey**: Checking whether an area is contaminated or not

3- **CHA**: Confirmed Hazardous Area, which is an area confirmed to be contaminated

4- **Clearance**: Cleaning a CHA using different demining techniques

5- **Cleared land**: An area that is no more contaminated after clearance process

6**- Completion report**: Finishing the report and setting the CHA as clean

1.3. Technologies Rising

The world is witnessing a big rise in two fields of technology. The first one is the Drones field where drones made us capable of reaching blind spots and angles we couldn’t see before. The other field is Machine Learning, which is a sub-area of AI that refers to the **"ability of computers to independently find solutions for specific problems by recognizing patterns in the input datasets"**. This technology has boosted the visual abilities of computers in the terms of extracting patterns from images and videos.

Combining drone technology with object detection (a machine learning technique for detecting objects from images) in a GIS environment would make us capable of analyzing images taken by the drone, auto detecting any object that a human eye can see, and saving the detected objects’ positions on a map.

1.4. Combining Technology with Demining

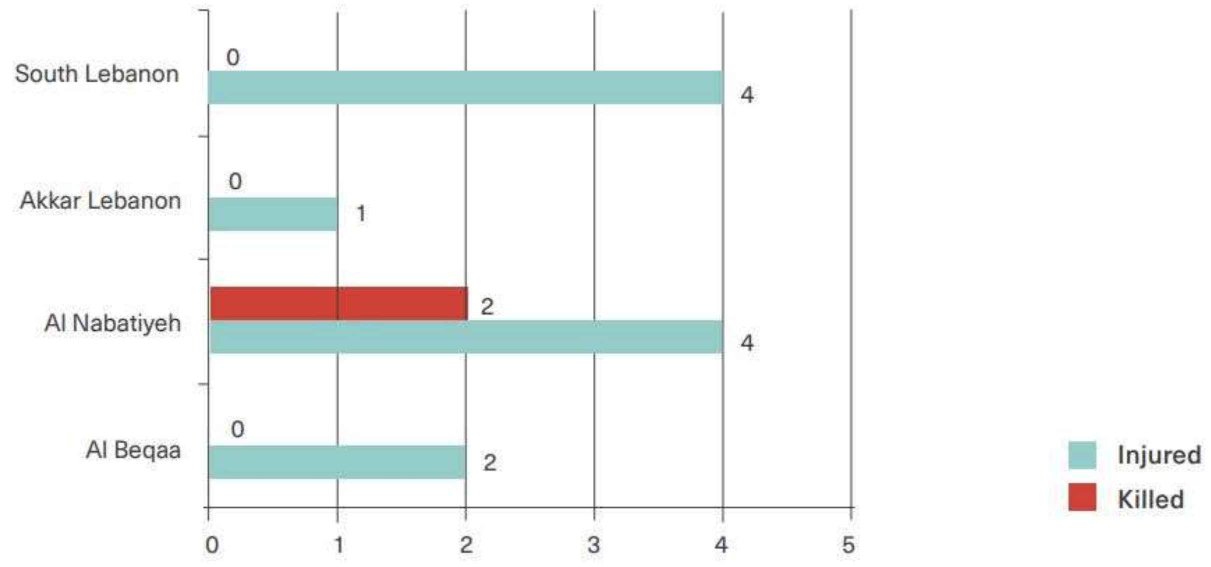
The idea of my project is developing a solution that enhances the demining process by the Lebanese army for the 'AO 2.5 RT' bombs found along the Lebanese-Syrian borders, where a drone captures multiple images covering the targeted area. Then an object detection model takes these images as an input and detects the position of the AO bombs in the images (if they exist). The output images will be processed by the **Drone2Map** software (by Esri) to create a map of the targeted area including all the bombs with buffers over the area in which each one is detected. Lastly all kinds of geospatial analysis are available.

The solution will give a clear view for the Lebanese Army officers about the rate of contamination of a certain area, and the overall distribution of bombs without the need of human approaching to the danger. This would result a remarkable boost in both **Land Survey** and **Clearance** steps of the demining workflow by telling whether a land is confirmed to be contaminated besides detecting most (not 100%) of cluster bombs found on the area of survey. Over 35% of the demining workflow will be covered by the project, saving lives, time, and resources.

1.5. Motivations

According to the annual report of LMAC, 13 mine victims resulting in 2 deaths including a young boy, and 11 casualties including 2 Syrians and 1 Palestinian were recorded in 2019.
A loss of cattle had also been recorded, which seems quite important with 13 fatalities and 3 casualties for 5 separate farmers and shepherds reflecting the socio-economic impact of mines.

AS SEEN IN THE FIGURE BELOW, THE HIGHEST NUMBER OF VICTIMS WERE IN SOUTH LEBANON.



The threat of mines on our society and economy combined with the hard work and remarkable efforts of the LMAC teams are the main reasons to develop this project and increase its efficiency over time.

2. State of Art

2.1 Introduction

When it comes to computer vision technologies, object detection has been one of the most trending topics through last years. However, aerial imagery field witnessed a slow development compared to other fields in the same domain because of 4 main difficulties:

- **Highly variant configurations:** where objects appear in different orientation, locations, and scales.

- **Objects size:** where target objects are usually very small in pixels with respect to the image dimensions.
- **Coupling constrains:** where training datasets contain objects in specific environments and backgrounds. A change in these variables may require training a new model over new datasets.
- **Resolution variety:** where objects in aerial images appear at a number of resolutions, from humans about 5 pixels wide to playgrounds 1000 pixels long.

2.2Previous Work

2.2.1 Public Datasets

Increasing the number of high quality datasets is one way to overcome some difficulties above. To achieve this purpose Gui-Song Xia et al. [1] created a "large-scale Dataset for Object deTection in Aerial images (DOTA)". They worked on aerial images of size about 4000*4000 pixels each, using 15 common object categories like cars, ships, harbors, etc.. . The output was 2806 annotated images containing 188,282 instances in every image. Such work enhances both quality and accuracy of the common object detection applications using aerial imagery.

2.2.1 Small clustered Objects

Many techniques are developed and used improve object detection over small objects and make it easy. Fan Yang et al. [2] worked on an effective method to detect clustered small objects in aerial imagery.
The idea of their work is detecting regions on images where targeted objects are clustered, and run their detection code over them instead of trying to detect single objects over the whole image.

The Cluster object Detection consists of 3 levels:
1- Detect the clusters of the input image using cluster proposal sub-network (CPNet)

2- Estimate object scales using a scale estimation sub-network (ScaleNet)
3- Detect objects using a dedicated detection network (DetectNet) on the scale-normalized cluster regions fed into it.
The result was reducing the number of chips needed for detection significantly and increasing the accuracy of detecting small objects effectively.

2.3 Our approach

Many other studies [3][4][5] had a major role enhancing the world of object detection and aerial imagery. However, in our case we deal with different kind of difficulties.
Here we try to detect very small non-clustered objects (with respect to the image size)that cannot be found in any public dataset like DOTA. Moreover, the bombs are designed in shape and color to resemble rocks in order to achieve camouflage and reduce the chance of detecting them. This adds a new challenge to our model knowing that our dataset contained only 20 images of the bomb at first! which is way far from enough.

Materials and Methods

The project workflow includes using different software and hardware tools to achieve different tasks like capturing images, training models, building maps, etc..
Here we report how the system was built in details, giving the used hardware and software, and how the work was done.

1. Hardware Used

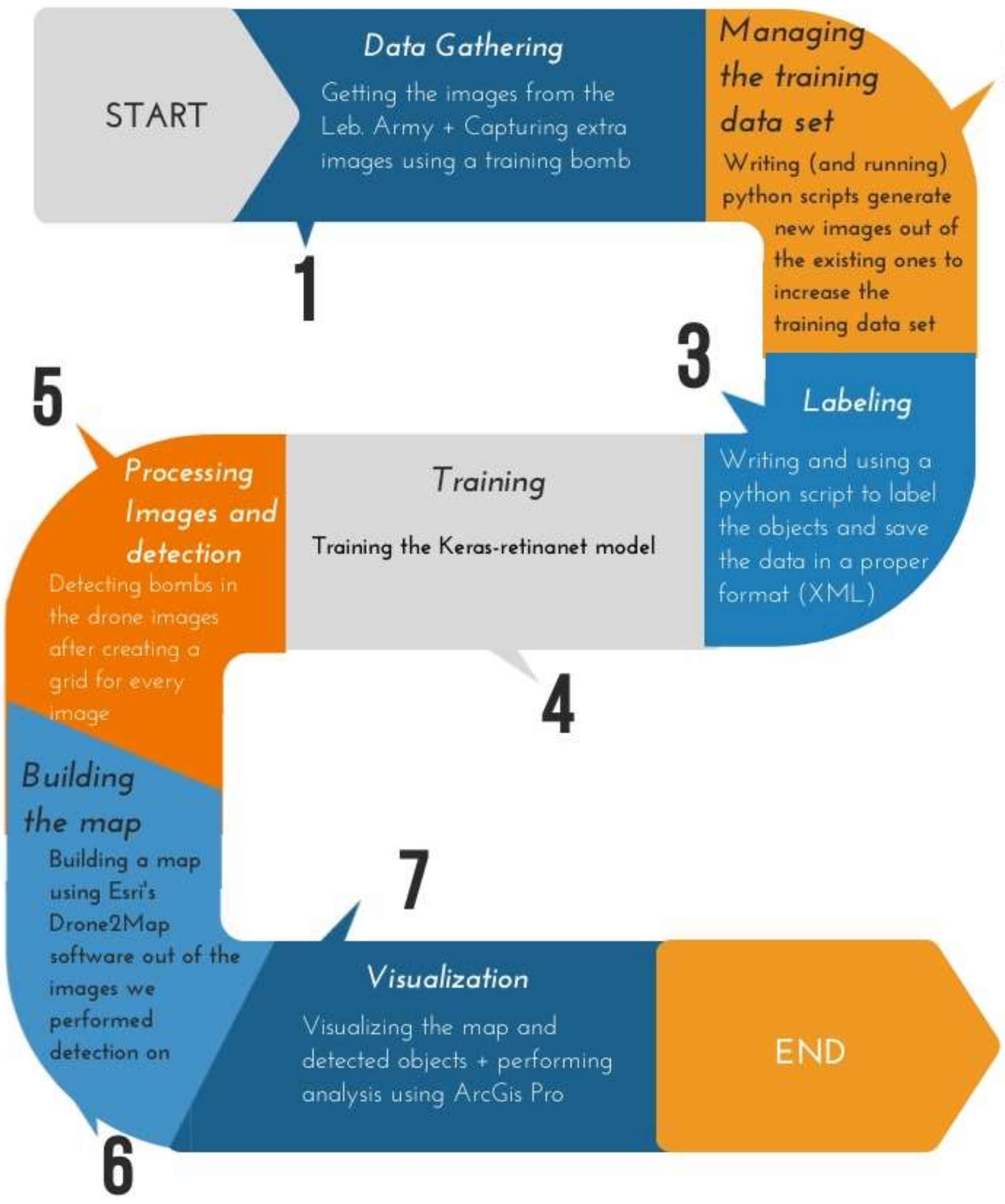
- DJI Mavic Pro drone

- A camera for collecting more images of the bomb prototype
- A laptop for developing scripts codes and labeling data
- A computer case for training the model and applying detection
- A Microsoft azure machine on the cloud for building the map

2. Software Needed

- Pycharm editor with Python 3.8
- ArcGIS Pro (from Esri)
- Drone2Map (from Esri)

3. Workflow of the Project


START
Data Gathering
Getting the images from the Leb. Army + Capturing extra images using a training bomb
1
Managing the training data set
Writing (and running) python scripts generate new images out of the existing ones to increase the training data set
2
3
Labeling
Writing and using a python script to label the objects and save the data in a proper format (XML)
Training
Training the Keras-retinanet model
4
5
Processing Images and detection
Detecting bombs in the drone images after creating a grid for every image
Building the map
Building a map using Esri's Drone2Map software out of the images we performed detection on
6
7
Visualization
Visualizing the map and detected objects + performing analysis using ArcGis Pro
END

3.1. Data Gathering

The first step of the project, was to get the data needed to build an accurate model capable of detecting cluster bombs in a difficult environment.

At first, we sent a permission request for the Lebanese Army to get the images they have for the AO bomb. After more than 5 weeks we received the images that were less than 20 images for the bombs! Such number is way far from enough for an object detection model that usually takes 1000+ input images to give high accuracy.

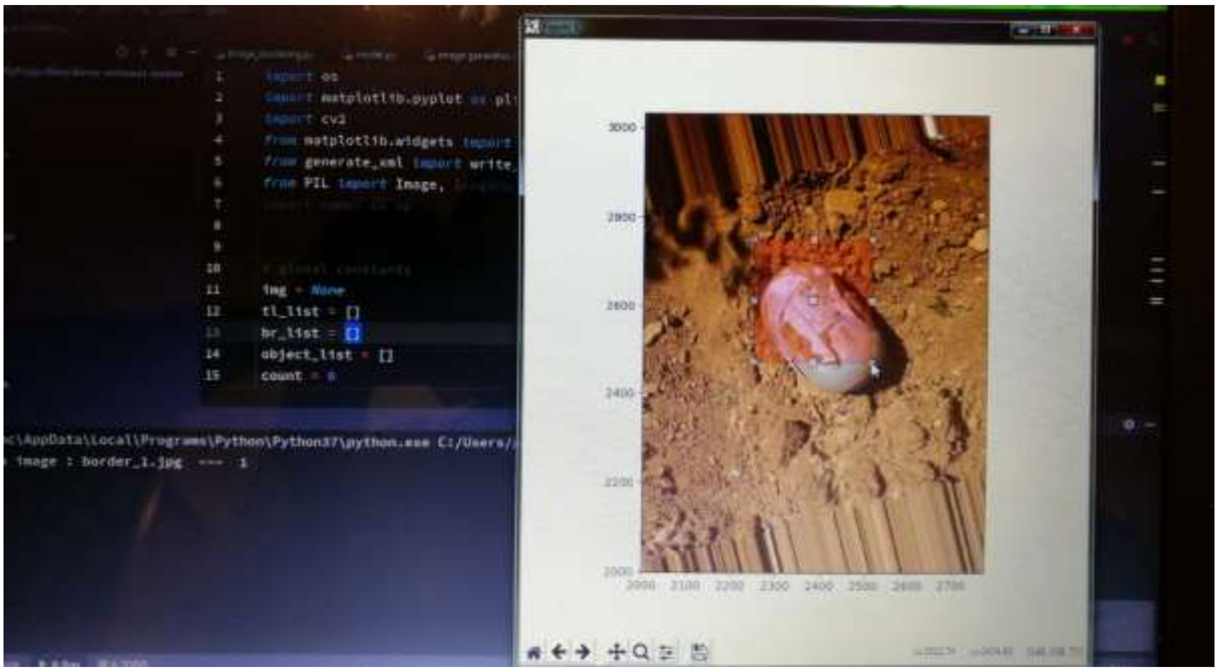
Since such data is classified as military, no datasets are available on the web, so we had to find alternative data sources to increase our dataset. The best choice was visiting the Army's training field in North Biqaa. The field has similar environment as the areas containing real AO bombs on the Lebanese-Syrian borders. Plus, it contains prototypes of almost all kinds of cluster bombs found in Lebanon, including the AO bomb we need. So we visited the field, took normal images of the bomb prototype, and took drone images of the whole training field including the bombs in it, in order to use them for testing our application.
The third step of data gathering is taking the bomb prototype with us to capture more images in similar/different environments.
After all the above, the number of training images in our hands reached174.

3.2. Managing the training data set

In the previous step we gathered and captured images with a total of 174 which is a nonsufficient number to lunch an object detection training with a high accuracy. That’s why we looked up a way to increase this number and we ended up by using a Python package called Image-Generator for editing images (brightness, orientation angles... etc) so that the number of images increased till 580.

3.3. Labeling

In order to label the 580 images, I wrote a Python script which allows me to label with squares. After that the images' coordinates are saved in a CSV and XML (PASCAL) format and were ready for training.



```
import os
import matplotlib.pyplot as pl
import cv2
from matplotlib.widgets import
from generate_xml import write_
from PIL import Image,



# global constants
img = None
tl_list = []
br_list = []
object_list = []
count = 0
```

```
c\AppData\Local\Programs\Python\Python37\python.exe C:/Users/
 image : border_1.jpg --- 1
```

Figure 1Drawing annotations using python script

3.4. Training

Regarding the training step, we had many options.

- ➢ The first option was to train using ArcGis Pro but it didn't work because this software is designed to train data labeled from satellite imagery using the software itself and not to train an external training data set.
- ➢ The other option was to train our own model using Python. We chose the Keras-Retinanet model for three reasons: it's simple, it has high accuracy even from a small data set and the third important reason is detailed in the following:
  Object detection architectures are split in two categories: single-stage and two-stage.

  Two-stage architectures first categorize potential objects in two classes: foreground or background. Then all foreground's potential objects are classified in more fine-grained classes: cats, dogs, cars, etc. This two-stage method is very slow but also, and of course, produces the best accuracy. The most famous two-stage architecture is Faster-RCNN.

On the other hand, single-stage architectures don’t have this pre-selection step of potential foreground objects. They are usually less accurate, but they are also faster. RetinaNet’s single-stage architecture is an exception: it reaches two-stage performance while having single-stage speed! [6]

3.5. Detection

Regarding detection, we had two options.

- The first option is to import the model resulted from training to ArcGis Pro and then apply detection using the software Pro on the drone images (drone images have latitude and longitude).

When we worked with the first option it turned out that there is limitation in ArcGis Pro in which it‘s exceptionally not designed to accept a Keras-Retinanet model.

- The Second option was to apply detection on drone images that we have after training. At first we tried to apply detection on the hole image but then we got high **False Positive** ratio and high **False Negative** ratio. The problem was that the bombs showing in the training data set were big with respect to the image size while bombs showing in the testing data set (drone imagery) were small with respect to the image size.

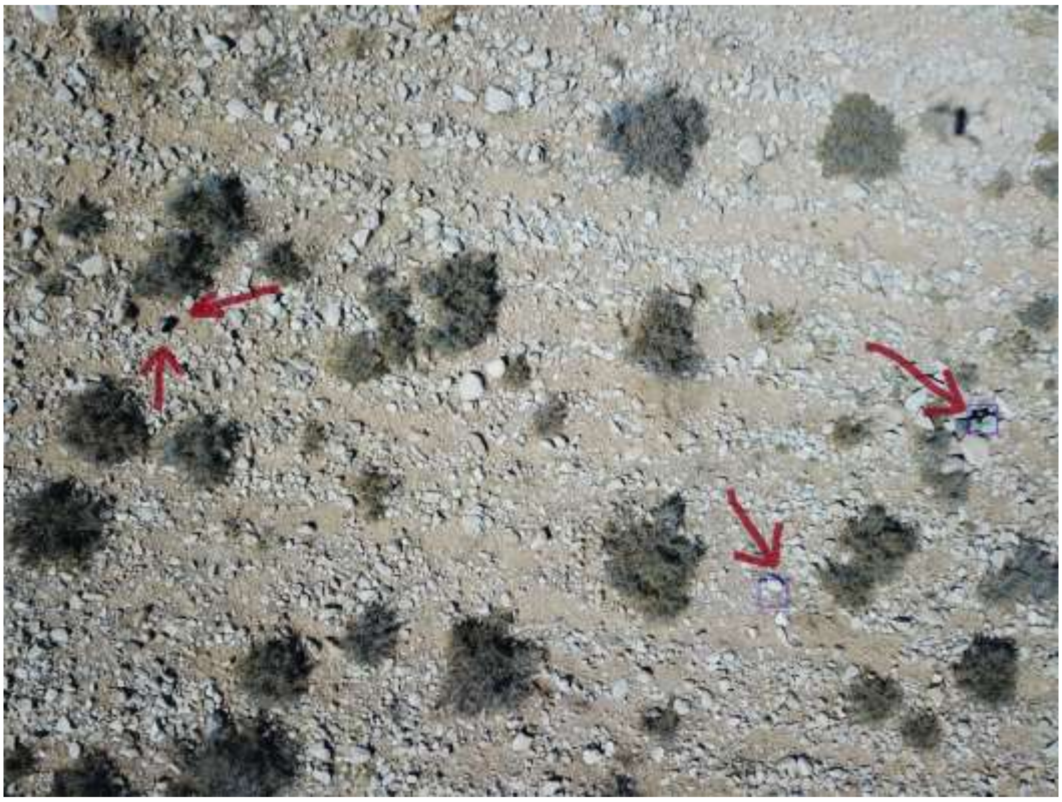


Figure 2 Undetected bomb to the left (False Negative) an two wrong detected objects to the right (False Positive)

To solve the problem, we had to add a grid on the image so that we can apply detection on each partition of the grid separately but in this case the metadata (latitude and longitude) of the image are being lost when gridding it. Another problem was that we had to find a way in order to show the object detected in a partition on the hole image.

Trying to solve the above error, we wrote a python script that does the following:

- Firstly, it takes a folder that contains the drone images that we need to apply detection on as an input and takes also the path of the model that we're going to apply detection by. It starts by roaming through the drone images that we will apply detection on. Meanwhile the size of the grid partition that we want and how much is the intersection between each partition must be defined.
- The second step and according to this given and these variables the Python code dynamically draws a grid (regardless of the image size) and applies detection on each partition and then records if any bombs were detected.

  In the case of a bomb detection the code translates x and y and then it draws a square around the bomb on the original image. Plus, it saves the coordinates of the original image in a text file. In this way we applied detection on the object in the original image using the object detection in the partition and the translation of x and y.

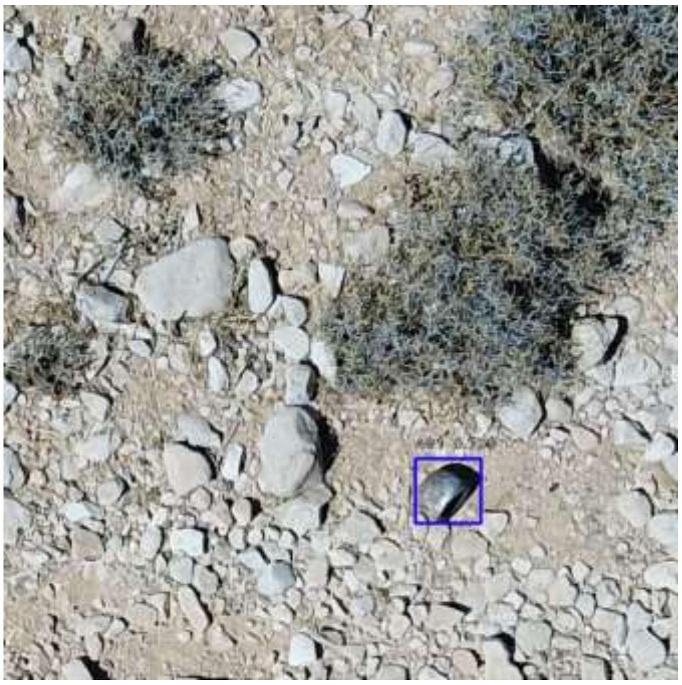


Figure 3A detected bomb from a partition of grid

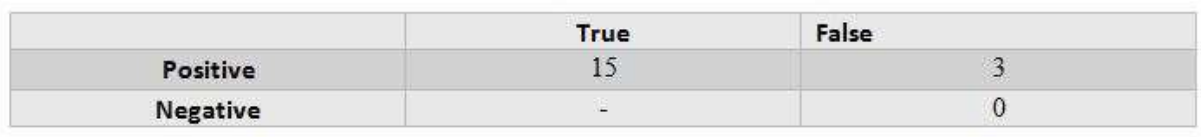
  The second problem mentioned above was that the metadata of the drone image that is being processed are being lost. This problem was solved by copying the metadata back to the original image from a backup copy using 2 python packages **Piexif** and **Gpsphoto**.

The output of this python script is a folder containing three things:

1) A sub-folder for each image, that contains all the partitions of the image's grid, with detection boxes on them.

2) Two text files (each with a different data format) that contain the longitude and latitude of each image with a detected bomb in it.
3) The drone images with detection boxes and its correct metadata.

After running the script over 74 drone images containing 15 bombs and hundreds of rocks having similar shapes, the results were as shown!



	True	False
Positive	15	3
Negative	-	0

Figure 4Results of performing detection over the drone images dataset

3.6. Building the map

After the bomb detection and the metadata copy, we used Esri’s software: Drone2Map to create a map out of the drone images we performed detection on.

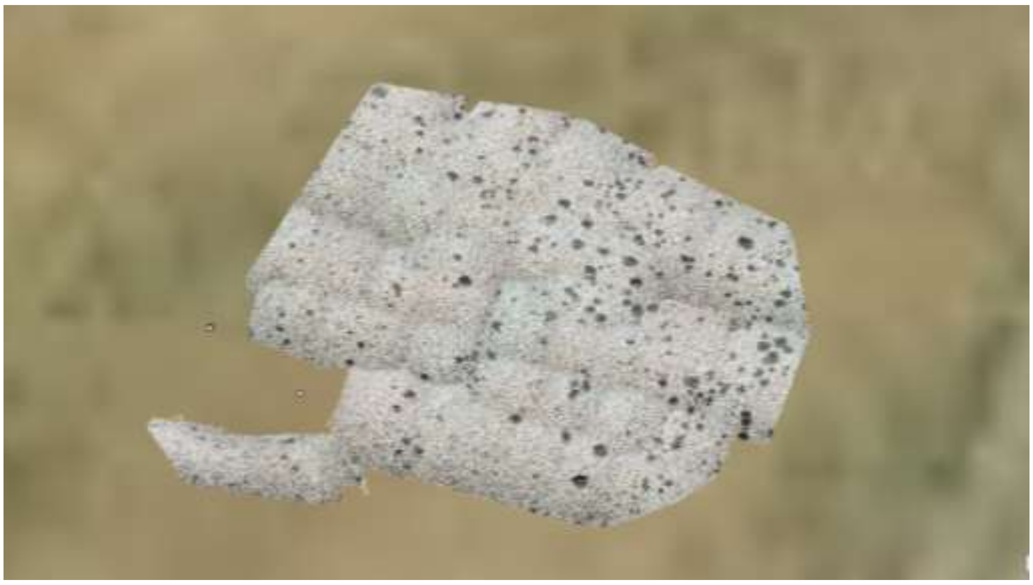
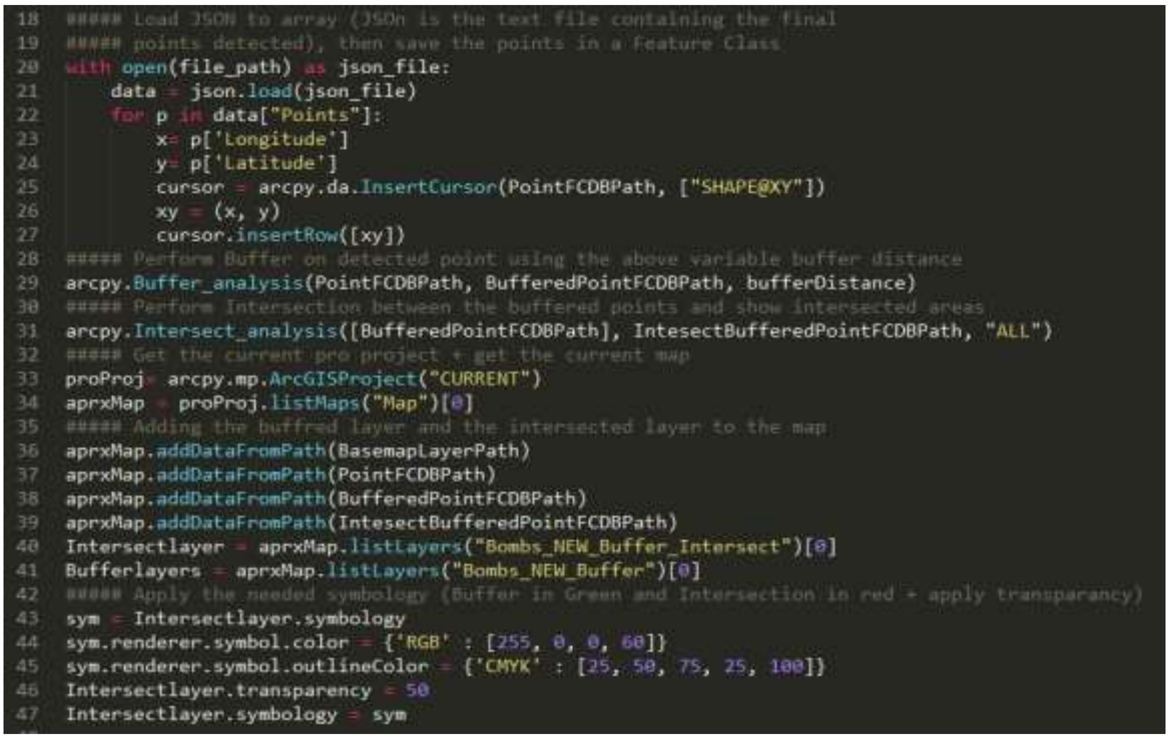


Figure 5The Map built using Drone2Map

3.7. Visualization



```
##### Load JSON to array (JSOn is the text file containing the final
##### points detected), then save the points in a Feature Class
with open(file_path) as json_file:
    data = json.load(json_file)
    for p in data["Points"]:
        x= p['Longitude']
        y= p['Latitude']
        cursor = arcpy.da.InsertCursor(PointFCDBPath, ["SHAPE@XY"])
        xy = (x, y)
        cursor.insertRow([xy])
##### Perform Buffer on detected point using the above variable buffer distance
arcpy.Buffer_analysis(PointFCDBPath, BufferedPointFCDBPath, bufferDistance)
##### Perform Intersection between the buffered points and show intersected areas
arcpy.Intersect_analysis([BufferedPointFCDBPath], IntesectBufferedPointFCDBPath, "ALL")
##### Get the current pro project + get the current map
proProj= arcpy.mp.ArcGISProject("CURRENT")
aprxMap = proProj.listMaps("Map")[0]
##### Adding the buffred layer and the intersected layer to the map
aprxMap.addDataFromPath(BasemapLayerPath)
aprxMap.addDataFromPath(PointFCDBPath)
aprxMap.addDataFromPath(BufferedPointFCDBPath)
aprxMap.addDataFromPath(IntesectBufferedPointFCDBPath)
Intersectlayer = aprxMap.listLayers("Bombs_NEW_Buffer_Intersect")[0]
Bufferlayers = aprxMap.listLayers("Bombs_NEW_Buffer")[0]
##### Apply the needed symbology (Buffer in Green and Intersection in red + apply transparancy)
sym = Intersectlayer.symbology
sym.renderer.symbol.color = {'RGB' : [255, 0, 0, 60]}
sym.renderer.symbol.outlineColor = {'CMYK' : [25, 50, 75, 25, 100]}
Intersectlayer.transparency = 50
Intersectlayer.symbology = sym
```

Figure 6Script of map importing and buffer creation tool

Visualization of results by creating 2 tools in the ArcGis Pro software that performs the following:

- Import the map that we created in the previous step.
- Make a feature class of the detected points out of the text file that was in the output of the script.
- Perform buffer on the detected points with a radius of 12 feet (which approximately covers the complete area of a drone image).

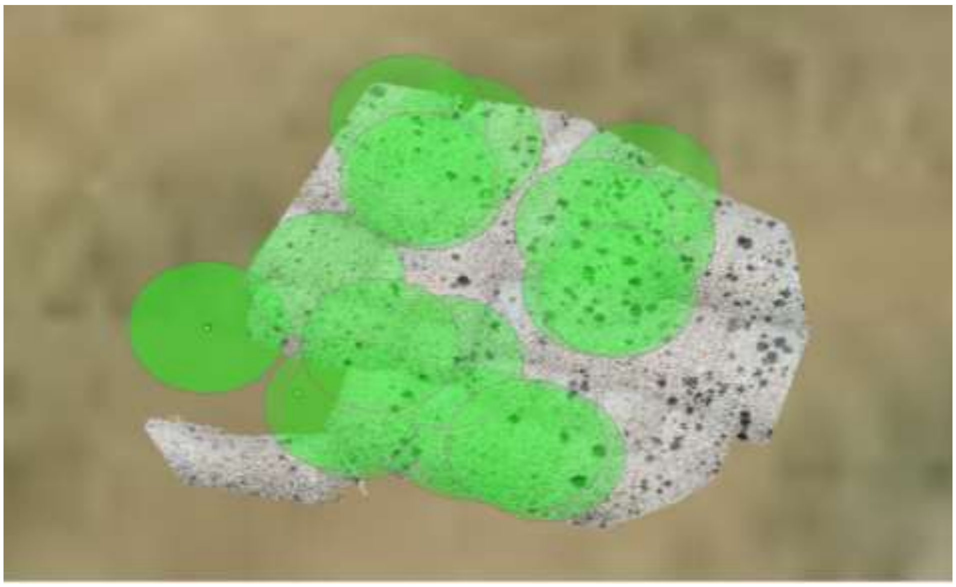


Figure 7Buffers applied on the map

- Perform an intersection for the buffers that was created so that we shrink the area where a bomb is detected since a single bomb could be detected in multiple drone images(due to intersection between them)
- Add the buffered layer and the intersection layer to the map.

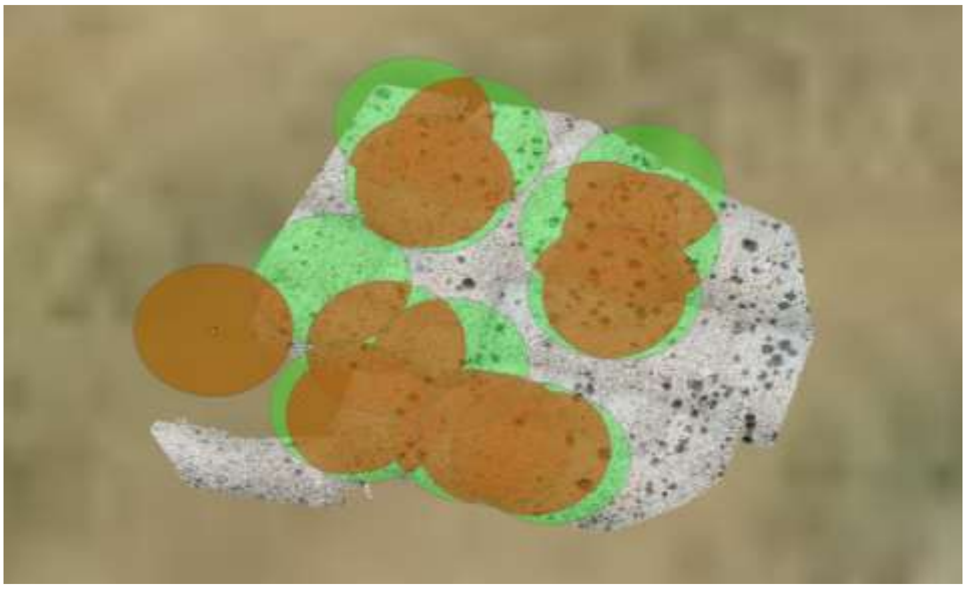


Figure 8Intersection of buffers applied to the map

- Apply the needed symbolization which is the color and the transparency of the buffers and intersections.

- The last step is to create a tool that turns the buffer and the intersection layer off and performs heat map over the result.

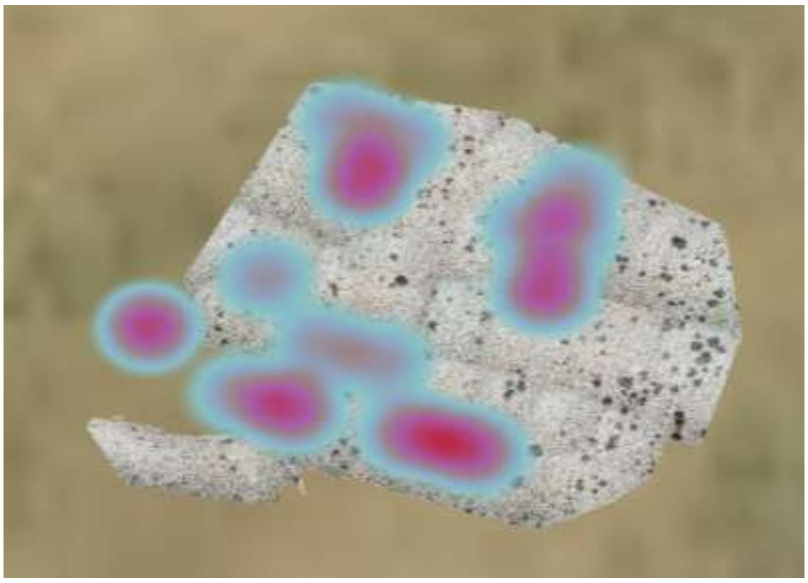
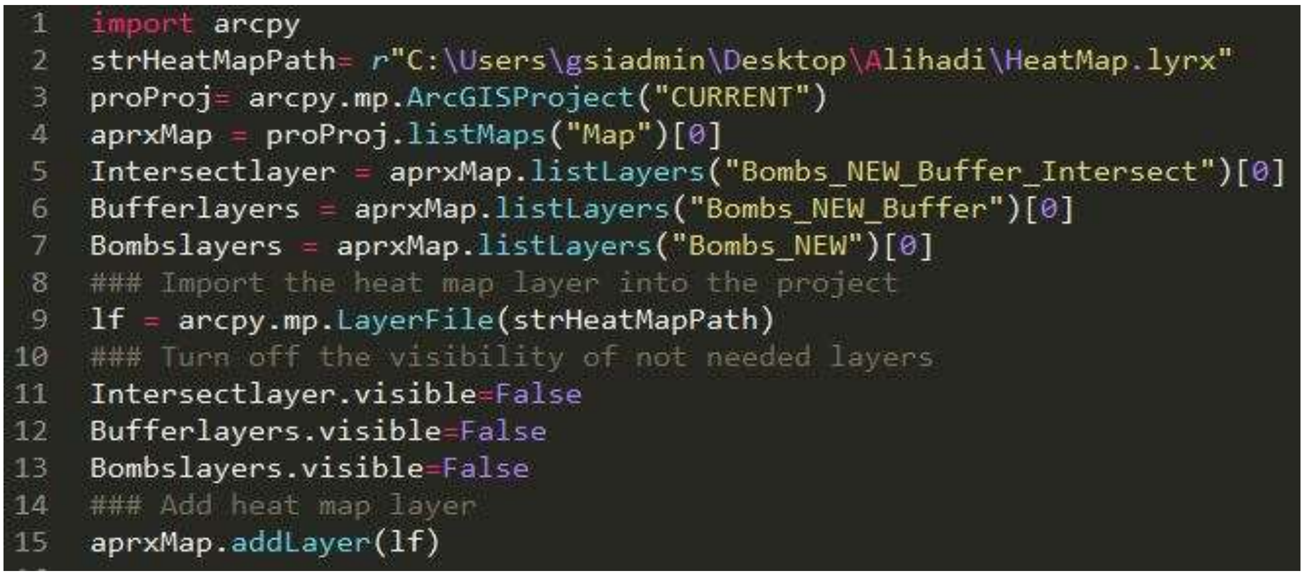


Figure 9Heat map applied to the map



```
import arcpy
strHeatMapPath= r"C:\Users\gsiadmin\Desktop\Alihadi\HeatMap.lyrx"
proProj= arcpy.mp.ArcGISProject("CURRENT")
aprxMap = proProj.listMaps("Map")[0]
Intersectlayer = aprxMap.listLayers("Bombs_NEW_Buffer_Intersect")[0]
Bufferlayers = aprxMap.listLayers("Bombs_NEW_Buffer")[0]
Bombslayers = aprxMap.listLayers("Bombs_NEW")[0]
### Import the heat map layer into the project
lf = arcpy.mp.LayerFile(strHeatMapPath)
### Turn off the visibility of not needed layers
Intersectlayer.visible=False
Bufferlayers.visible=False
Bombslayers.visible=False
### Add heat map layer
aprxMap.addLayer(lf)
```

Figure 10Script of heat map tool

The final result is a map including a feature class containing the positions if the areas where a bombs were detected, and feature layers representing the 12 feet buffers around the points, the intersection of the buffers, and a heat map.

Note: All the variable controlling the radii and colors of the buffers and heat map can be easily modified .

Conclusion and Future Work

After the huge effect of AI and GIS technologies in our lives, we wanted to benefit from them to help our country and army. These technologies provide big potentials for image processing and locations-based applications. However, we had multiple limitations that turned our project into a challenge. So we had to be creative and find our own solutions that may help us in similar projects.

The main purpose of this project is to enable scanning large areas that possibly contain bombs without the need of any human approach near the danger zone. Although the 100% certainty of detecting all the bombs in a target area is hard to achieve, we can still enhance the demining process by showing the positions of the detected bombs in a clear map where all analysis tools can be applied.

The successful result using limited resources proves that further development can take the demining process to another level.

1. As demining and object detection actions go on, more data will be gathered, and more data is the key for higher accuracy by re-training the model over and over.
2. Different drones with different sensors (heat sensor, waves other than RGB ...) can be used to combine the results and detect bombs that cannot be detected using a normal camera.
3. Detecting objects other than the 'AO 2.5 RT' bomb could be done with similar results if the data is available.
4. Models other than keras-retinanet may also be useful in our case. We can test more models and compare the results to choose the one with the best performance in the future. Some examples of models to test: Fast RCNN, SpaceNet, etc..
5. The drone imagery we used were taken manually which caused many intersections between the buffers. However, Esri announced a new product that may solve the problem. It is called **Site Scan for ArcGis** which provides an end-to-end workflow for flight planning and acquiring images.

6. The time needed for creating the grids and detecting each partition is slightly long, which can be made shorter by more code-optimization.

Conclusion and Future Work

[1] DOTA: A Large-scale Dataset for Object Detection in Aerial Images∗ Gui-Song Xia1† , Xiang Bai2† , Jian Ding1 , Zhen Zhu2 , Serge Belongie3 , Jiebo Luo4 , Mihai Datcu5 , Marcello Pelillo6 , Liangpei Zhang1 1Wuhan University, 2Huazhong Univ. Sci. and Tech. 3Cornell University, 4Rochester University, 5German Aerospace Center (DLR), 6University of Venice

[2] Clustered Object Detection in Aerial Images Fan Yang1 Heng Fan1 Peng Chu1 Erik Blasch2 Haibin Ling3,1∗ 1Department of Computer and Information Sciences, Temple University, Philadelphia, USA 2Air Force Research Lab, USA 3Department Computer Science, Stony Brook University, Stony Brook, NY, USA

[3] K. Liu and G. Mattyus. Fast multiclass vehicle detec- ´ tion on aerial images. IEEE Geosci. Remote Sensing Lett., 12(9):1938–1942, 2015

[4] S. Razakarivony and F. Jurie. Vehicle detection in aerial imagery: A small target detection benchmark. J Vis. Commun. Image R., 34:187–203, 2016.

[5] Y. Long, Y. Gong, Z. Xiao, and Q. Liu. Accurate object localization in remote sensing images based on convolutional neural networks. IEEE Trans. Geosci. Remote Sens., 55(5):2486–2498, 2017.

[6] https://medium.com/data-from-the-trenches/object-detection-with-deep-learning-on-aerial-imagery-2465078db8a9