DRONE DETECTION AND TRACKING WITH SUPER-RESOLUTION

A PROJECT REPORT

Submitted in partial fulfillment of the
Requirement for the award of the
Degree of

BACHELOR OF TECHNOLOGY
IN
ELECTRONICS AND COMMUNICATION ENGINEERING

by

Sai Krishna M (18BEC1270)

Under the Guidance of

Dr. Vinayak Nageli Shriniwas (External Guide)
Dr. Idayachandran G. (Internal Guide)

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRONICS ENGINEERING
VELLORE INSTITUTE OF TECHNOLOGY
CHENNAI - 600127

April 2022

CERTIFICATE

This is to certify that the Project work titled "***Drone detection and tracking with Super Resolution***" that is being submitted by ***Sai Krishna M*** **(18BEC1270)** is in partial fulfilment of the requirements for the award of **Bachelor of Technology in Electronics and Communication Engineering**, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

Idayachandran

Dr. Idayachandran G
Guide

The Project Report is satisfactory

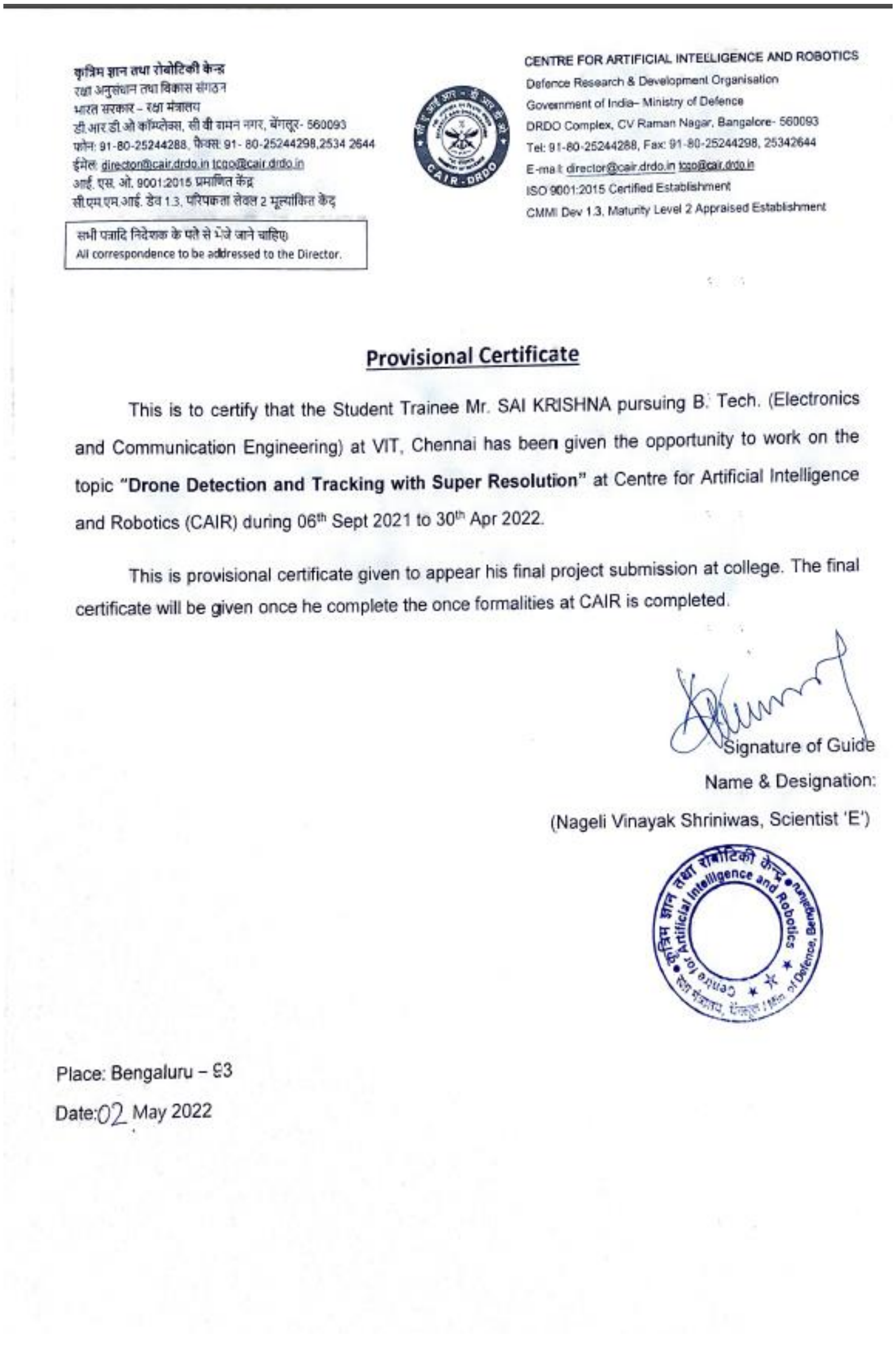
Internal Examiner **External Examiner**

Approved by

Head of the Department
B. Tech. (ECE)

DEAN
School of Electronics Engineering

BONAFIDE CERTIFICATE



कृत्रिम ज्ञान तथा रोबोटिकी केन्द्र
रक्षा अनुसंधान तथा विकास संगठन
भारत सरकार – रक्षा मंत्रालय
डी.आर.डी.ओ कॉम्प्लेक्स, सी वी रामन नगर, बेंगलूर- 560093
फोन: 91-80-25244288, फैक्स: 91- 80-25244298,2534 2644
ईमेल: direction@cair.drdo.in tcgo@cair.drdo.in
आई. एस. ओ. 9001:2015 प्रमाणित केंद्र
सी.एम.एम.आई. डेव 1.3, परिपकता लेवल 2 मूल्यांकित केंद्

सभी पत्रादि निदेशक के पते से भेजे जाने चाहिए।
All correspondence to be addressed to the Director.

CENTRE FOR ARTIFICIAL INTELLIGENCE AND ROBOTICS
Defence Research & Development Organisation
Government of India– Ministry of Defence
DRDO Complex, CV Raman Nagar, Bangalore- 560093
Tel: 91-80-25244288, Fax: 91-80-25244298, 25342644
E-mail: director@cair.drdo.in tcgo@cair.drdo.in
ISO 9001:2015 Certified Establishment
CMMI Dev 1.3, Maturity Level 2 Appraised Establishment

CAIR - DRDO

Provisional Certificate

This is to certify that the Student Trainee Mr. SAI KRISHNA pursuing B. Tech. (Electronics and Communication Engineering) at VIT, Chennai has been given the opportunity to work on the topic **"Drone Detection and Tracking with Super Resolution"** at Centre for Artificial Intelligence and Robotics (CAIR) during 06th Sept 2021 to 30th Apr 2022.

This is provisional certificate given to appear his final project submission at college. The final certificate will be given once he complete the once formalities at CAIR is completed.

Signature of Guide
Name & Designation:
(Nageli Vinayak Shriniwas, Scientist 'E')

कृत्रिम ज्ञान तथा रोबोटिकी केन्द्र
Centre for Artificial Intelligence and Robotics
रक्षा मंत्रालय, बेंगलूर / Min. of Defence, Bengaluru

Place: Bengaluru – 93
Date:02 May 2022

ACKNOWLEDGEMENT

I would like to thank my external guide Dr. Vinayak Nageli Shriniwas and the whole of CAIR, under whose guidance I carried out the project. I would like to thank my internal guide Dr. Idayachandran G sir for helping and guiding me through the course of the project. We would also like to thank our SENSE department head Dr. Vetrivelan P, dean Dr. Sivasubramaniam A for giving us this opportunity.

Sai Krishna

Sai Krishna M
(18BEC1270)

ABSTRACT

Drone detection as a technology has multiple uses in the automated world. It involves the detection of drone in an image or video frame using image processing techniques and algorithms. Drones are being used in various industries nowadays, ranging from delivery of goods and medicines, to military purposes like scoping and surveillance. This leads to the requirement of supervising them in order to make sure they perform as intended.

The current choice for such detection tasks is using Convolutional Neural Networks, and Deep Learning for automated detection with least manual support. But one of the difficulties of drone detection is that the size of the target object is almost always significantly small in ratio as compared to the size of the image. This leads to missed detections, because the algorithm isn’t always equipped enough to identify drones of such small size, owing to how convolutional neural networks work on an image. To tackle this issue, we explore the usage of Super Resolution networks in drone detection, and subsequently in tracking.

In this project, we experiment with and assess the impact of ESRGAN, a super resolution network, on drone detection using YOLOv4 detection network. We experiment with selective search for region proposal, and with tile-based detection as ways to include super resolution into the pipeline. We then proceed to implement our super resolution-based detection network into a correlation tracker and observe the output.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVATIONS

LIST OF ABBREVIATION

ABBREVATION	TITLE	PAGE NO.
CAIR	Centre for Artificial Intelligence & Robotics	1
DRDO	Defence Research & Development Organisation	1
SRGAN	Super Resolution – Generative Adversarial Network	1
SR	Super Resolution	1
YOLO	You Only Look Once (Algorithm)	1
mAP	Minimum Average Precision	2
CNN	Convolutional Neural Network	2
RCNN	Regions with Convolutional Neural Networks	2
DBPN	Deep Back-Propagation Network	2
SSD	Single Shot Detector	7
LR	Low Resolution	7
HR	High Resolution	7
SISR	Single Image Super Resolution	7
MISR	Multiple Image Super Resolution	7
GAN	Generative Adversarial Networks	8
PSNR	Peak Signal-to-Noise Ratio	9
SSIM	Structural Similarity Index Metric	9
MSE	Mean Squared Error	10
VGG	Visual Geometry Group	10
RGB	Red Green Blue (Channels)	12
RRDB	Residual in Residual Dense Block	12
ReLU	Rectified Linear activation Unit	12
ESRGAN	Enhanced SRGAN	12
API	Application Programming Interface	14
ROI	Region of Interest	19
NMS	Non-Maximum Suppression	22
MOSSE	Minimum Output Sum of Squared Error	25

CHAPTER 1
INTRODUCTION

The Centre for Artificial Intelligence and Robotics (CAIR) division of Defence Research & Development Organisation (DRDO) focuses on developing intelligent systems for secure communication, command and control, in the defence sector. Their mission is as follows:

"To add value to information by delivering dependable information systems to the Defence services for battle space dominance, by developing domain & technologies that ensure relevance, security, safety, resiliency, survivability and trustworthiness enabling the use of these systems in mission critical applications with required guaranties of assured performance"

As an intern at this institution, I was drafted to work on Deep Learning based drone detection and tracking assisted by Super Resolution. We experimented with SRGAN and YOLOV4 for super resolution and drone detection, by implementing them in various ways for better impact on the detection results. We also implemented the same on a correlation filter-based tracker and observed the results.

1.1 Objectives

The following are the objectives of this project:

- In this project, we aim to analyse how the Super-Resolution (SR) technique, by which images can be enhanced in resolution, can affect results in detection of drones.
- We are to compare and contrast the results of drone detection, with and without the assistance of Super Resolution.
- We experiment with ways in which super resolution can be brought into the detection pipeline, and how it affects our detection results.
- Subsequently, we implement the Super-Resolution algorithm into the drone tracking pipeline.

1.2 Background and Literature Survey

Drone detection and tracking becomes a necessary task in most places where drones are utilized. However, drone detection is not trivial and should not be compared with the detection of other objects. Realistically speaking, the size of a drone is almost always significantly small as compared to the size of the frame of the camera being used, due to the drone being at a distance. Thus, there are too many negatives in the image as compared to the positives, being the drones. This overwhelms the detector, and causes it to mis-detect objects to be drones, or miss the drones in the frame in many cases. Due to this, even the most powerful deep learning-based detection methods suffer from reduced accuracy in detection, and subsequent difficulty in tracking. One solution would always be to develop more complex networks that can handle this problem of scale, but the deployment of such networks is questionable on most edge devices.

Under these circumstances, super resolution is being researched and used to enhance the scale of the image or parts of the image, and thus enhancing the scale of the object, to make detection of such small objects easier. Depending on the algorithm used, applying super resolution to the whole image may not be a viable solution. As the size of the frame increases, the amount of compute and time required to enhance the image increases as well, and this may not be practically feasible. And this especially holds true for tracking assignments as well. Thus, it becomes imperative that there must be other ways in which super resolution can be applied in a non-intensive manner, to get the best possible results out of it.

Table 1.1 Literature Survey

Sl. No	Name of article	Authors	Information included
1.	Automated Drone Detection Using YOLOv4	Subroto Singha, Burchan Aydin	Drone detection was done using the YOLOv4 model on the Drone vs Bird dataset. They achieve an mAP of 74.36%.
2.	Real-Time Drone Detection Using Deep Learning Approach	Manjia Wu, Weige Xie, Xiufang Shi, Panyu Shao, Zhiguo Shi	They use a modified version of the YOLOv2 network on the USC drone video dataset, and achieve a precision rate of 88.35%
3.	Drone Detection in Long-range	Mrunalini Nalamati,	They experiment with

	Surveillance Videos	Ankit Kapoor, Muhammed Saqib, Nabin Sharma, Michael Blumenstein	various CNN backbone and detector architectures for small-drone detection.
4.	A deep learning approach to drone monitoring	Yueru Chen, Pranav Aggarwal; Jongmoo Choi; C.-C. Jay Kuo	Paper uses MDNet tracker with Faster-RCNN backbone to detect and track drones. They also construct their own dataset by various augmentation methods applied to the USC drone dataset.
5.	Task-Driven Super Resolution: Object Detection in Low-Resolution Images	Muhammad Haris, Greg Shakhnarovich, Norimichi Ukita	Showcases the results of using DBPN (Deep Back-projection networks for SR) network on the PASCAL VOC dataset, to show how super resolution can impact the results of object detection. They also explore the impact on blurred images, noisy images, etc.
6	The Effects of Super-Resolution on Object Detection Performance in Satellite Imagery	Shermeyer J, Van Etten A	Illustrates the impact of using super resolution for small object detection in satellite imagery.

1.3 Organization of the Report

The remaining chapters of the project report are described as follows:

- Chapter 2 contains the theory behind drone detection and super resolution algorithms.
- Chapter 3 contains the implementation of the super solution and detection algorithms, and the observations made.
- Chapter 5 compiles the results obtained after the project was implemented.
- Chapter 6 concludes the report with discussions about the results obtained and their future implications.

CHAPTER 2

THEORY BEHIND DRONE DETECTION AND SUPER RESOLUTION

This Chapter describes the theory, methodology, and design of the project.

2.1 Methodology

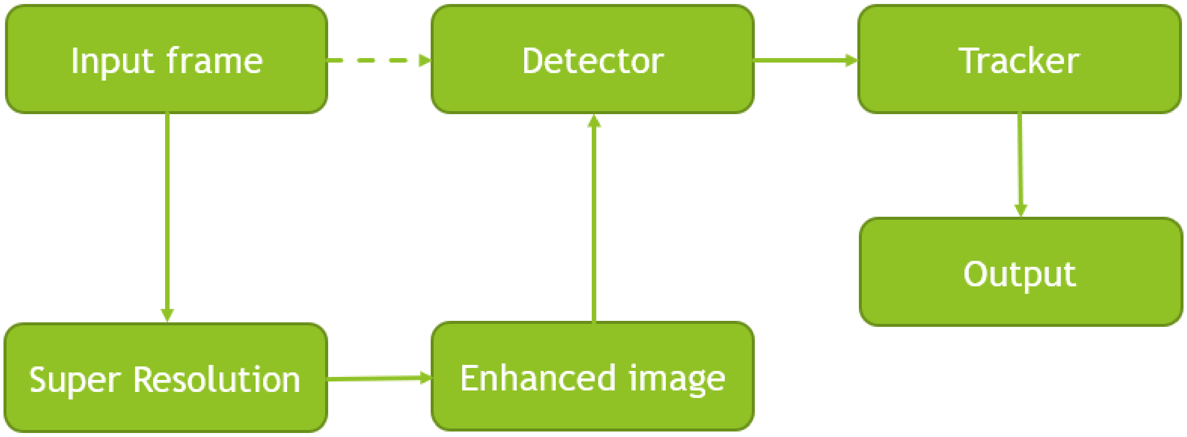


Figure 2.1: Overall Project Flow

The conventional pipeline for a generic tracker would be to give the current frame/image to a detector, which would generate detections as the inputs for the tracker algorithm, which would then associate different detections of the respective objects over time and track them. In this project however, we attempt to introduce Super Resolution into the pipeline as a pre-processing step before detection.

The image would then be enhanced in resolution by the super resolution algorithm, and the enhanced image would then be given to the detector. How the image is enhanced is something to be given thought. In this project, we seek to try to enhance the entire image, enhance selective regions of the image where the drone may be located, divide the image into patches and enhance and perform detection on the patches individually, etc. Once this is done, we feed the detection results to the tracker, which begins tracking the drone. In this project, we went with a correlation filter-based tracker for the job.

2.2 Convolutional Neural Networks as object classifiers

Convolutional neural networks have become the go-to approach when it comes to tasks like object classification and detection, image generation, etc. Since the first LeNet5 network by Yann LeCun in 1994, the capabilities of this class of algorithms has improved by leaps and bounds. Before such networks came about, hand-crafted feature extractors were employed to extract certain generic features from the image to make inferences. But this process is slow and does not have much scope. While such feature extractors may be used to find edges or such patterns in an image, it cannot do any more complicated tasks to a satisfiable accuracy. And this is especially true if you consider images more than 32x32 pixels in resolution. Images of these days are thousands of pixels in width and height, and require much more complex networks for processing.

Neural Nets are complex learning algorithms which try to learn a simpler, purpose-oriented representation of the data given to them. Convolutional neural networks take it one step further and employ convolution to focus on certain aspects of the image, thus making it viable to use for image classification and object detection tasks. In essence, a neural network is basically a model full of tuneable parameters which can learn complex representations. Any neural network has two stages, primarily: Training, and testing. In the training stage, training data consisting of the inputs and expected outputs is fed into the network, and the network learns the patterns in the data which can help it to achieve the expected output.

A convolutional neural network contains two parts: the convolutional layers in the front and the fully connected (a.k.a. Dense) layers in the back.

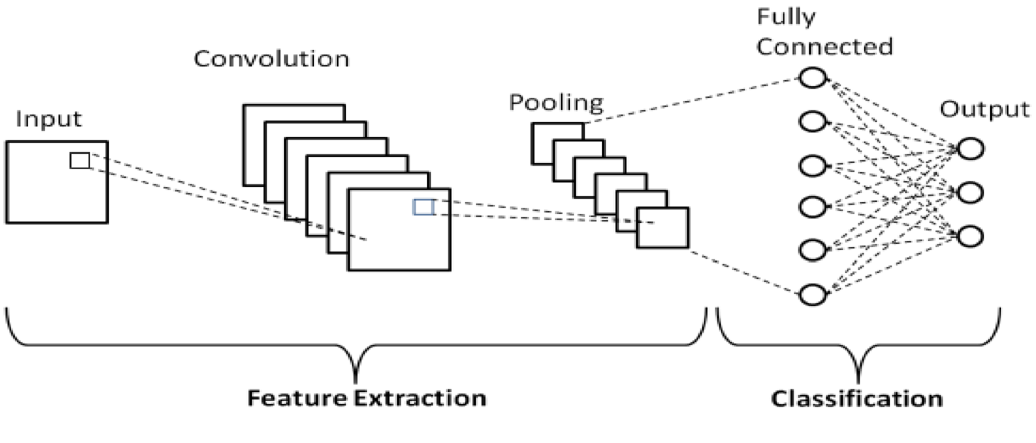


Figure 2.2: A simple ConvNet

The figure above illustrates the basic structure of a CNN. The input image is passed through several conv-blocks, which are a set of convolutional filter layers and pooling layers, stacked together. This is the part where features from the input image are extracted, and then passed onto fully connected Dense layers which are responsible for making the final decision.

In the training phase, the error generated by the output of the network is propagated back into the network, and is used to tweak and tune the weights of the convolutional filters used and any other tune-able hyperparameters in the network architecture. This allows for the network to learn to extract sophisticated features from the image pertaining to the task it is performing. For example, to classify between dogs and cats, the network learns to distinguish between cat and dog ears and mouths.

One interesting fact about CNNs is that the early layers extract low level features from the image and the complexity of the features extracted will increase as the depth of the network grows. And this is why deeper networks are capable of learning intricate features more than shallow networks.

2.3 Object detection using Convolutional Neural Networks

Now, for object detection, the network has to learn to classify the object in the image as well as localize it properly enough to put a bounding box around the object. This means that the network has to learn to predict the coordinates of the box around the object as well as the class.

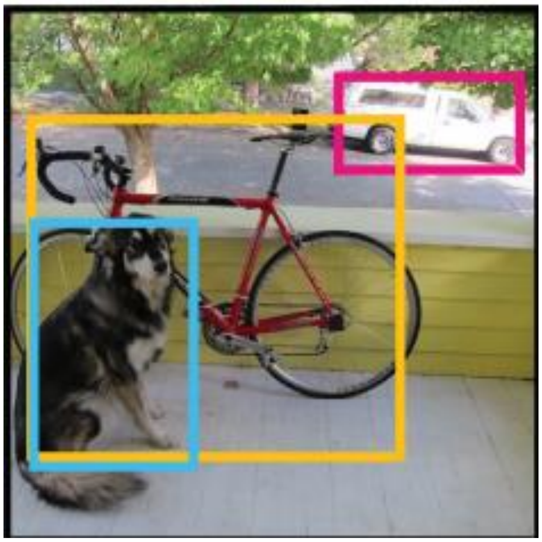


Figure 2.3: Object Detection [14]

Various object detection algorithms exist in the market, but the ones that involve convolutional neural networks prove to be the most popular ones. Popular deep-learning based detectors include YOLO, MobileNet-SSD, and RCNN.

One important categorisation among object detectors are single stage and multi-stage object detectors. Multi-stage detectors such as the RCNN family run the image through multiple stages of processing before making the detections. RCNN in particular performs region proposal before detection, wherein the algorithm identifies regions of interest (ROIs) in the image that may contain an object. Then, a classifier is run on each of those patches and detections are thus made. This would appear to be an obvious method of choice for object detection, but it comes with a caveat. Since it involves two stages, the algorithm tends to be slow and impractical at times for real-time usage. In this project, we have also experimented with the selective search algorithm in section 3.3.

Single stage detectors, as opposed to multi-stage ones, only make a single pass through the image and make the detections based on that singular pass. The YOLO family is one such popular class of algorithms wherein the image is divided into squares and the detector is run individually on each square, after which all the detections are collectively pooled together. This tends to be faster than many two-stage detectors, but do lack in accuracy a bit. But they are found to be satisfyingly accurate and fast enough to be used for real-time purposes. In this project, we have used a YOLOv4 [8] drone detector trained on drone images.

2.4 Super Resolution

Super Resolution (SR) refers to a class of image (or video) processing algorithms used to enhance the resolution of an image (or video) with minimum effect on quality. Specifically, we aim to get High Resolution (HR) images from Low Resolution (LR) images. It is assumed that an LR image is an HR image which underwent degradation.

Specifically for this project, we focus on Single Image Super Resolution (SISR), where we focus on enhancing a single image and using only the data provided in the image itself. The other type of Super Resolution, Multi-Image Super Resolution (MISR), we employ multiple images of the same scene taken from different spatial and temporal points and use them all collectively to generate a high-resolution image of the scene.

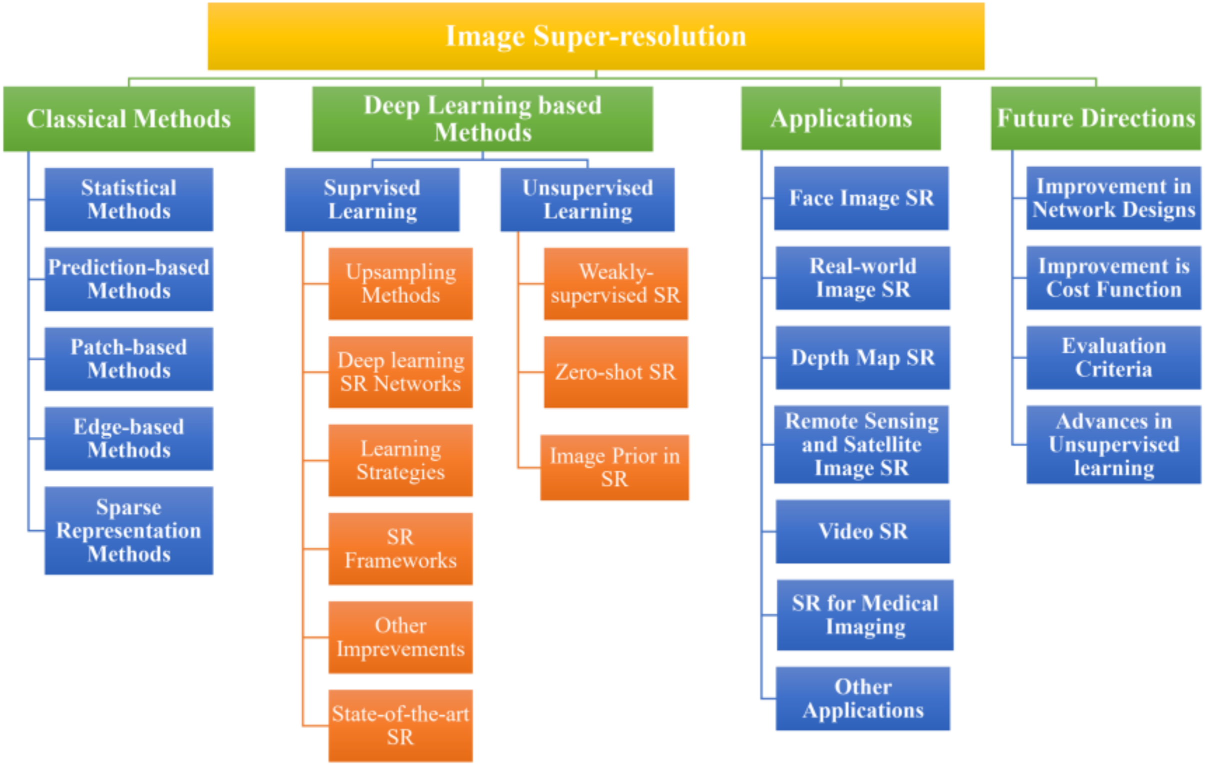


Figure 2.4: Methods of Image Super Resolution [13]

Deep Learning based SISR algorithms have been researched on from 2014, when the first deep convolutional network was used for image super resolution. Currently, networks known as Generative Adversarial Networks (GANs) are extensively used for Image Super Resolution and have achieved considerable success in this field. But these aren't the only methods used for super resolution. Several conventional methods for super resolution exist, which do not make use of Deep Learning for the task. Such methods are called as classical methods of Super Resolution. One of the first approaches was based on Lanczos filtering, wherein a scaled image was sharpened using this filter. Various other approaches like edge-based methods, wherein filters are used to smoothen the edges in an image to make it look more realistic after upscaling.

But conventional methods were very sophisticated and subjective, and deep learning-based methods soon replaced them. These were broadly divided into supervised and unsupervised methods. Supervised methods are trained on large datasets involving LR-HR image pairs. This begs a question, what should be the loss function for training super resolution networks? How do we grade the output of the network? Early works used pixel loss, which involved measuring the error in reconstruction of the image in higher resolution. But as more research was done, it was found that pixel loss was not suitable enough for this task. As algorithms evolved, a variety of loss functions such as adversarial loss and content loss were found to work better. These became ever so important with the advent of GANs into this field as well.

But how do we grade a network as doing well? The loss function used by a network is primarily for optimizing the network and not for grading the quality of the output. For comparison of quality, multiple objective and subjective metrics are being used across research. Peak Signal-to-Noise Ratio (PSNR) is one such metric being used for this purpose. Structural similarity using Structural Similarity Index Metric (SSIM) is also a commonly used objective metric. Then again with the nature of this task, a major part of research also takes into account certain subjective metrics. This was primarily because objective metrics were found to fail in certain circumstances. Even when the pixel reconstruction loss was minimal, the image as such was found to be of low quality to the naked eye. This is articulated well in the below picture.

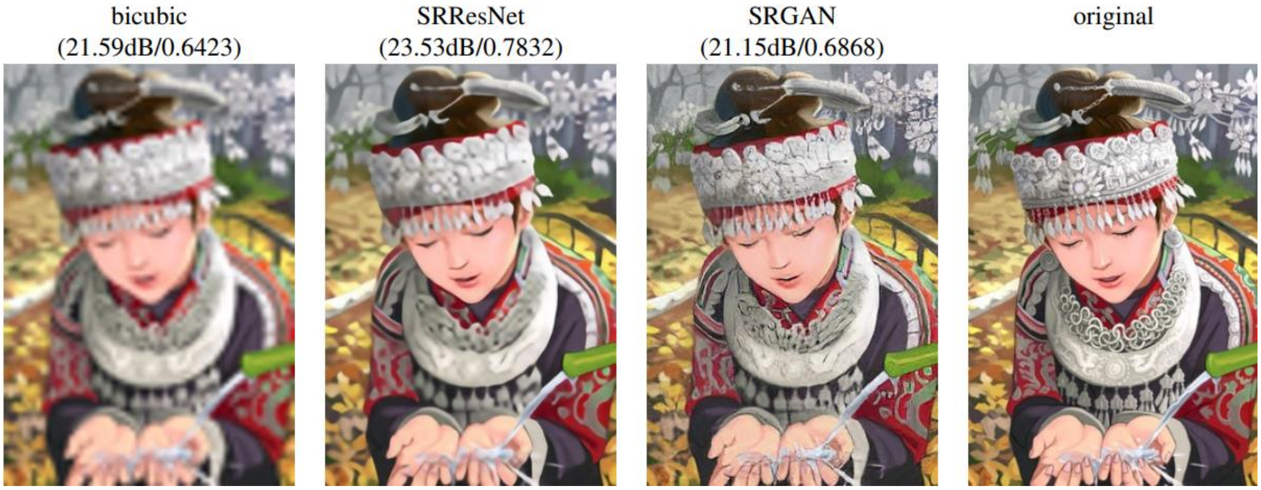


Figure 2.5: Super Resolution sample output [4]

In figure 2.5, the SRResNet output was graded to be of higher PSNR than the SRGAN output. But clearly, the latter is of higher quality than the former with respect to texture quality and detail. Thus, subjective metrics like perceptual quality and opinion scoring came into the norm.

2.4.1 GANs for Super Resolution and SR-GAN

In a Generative Adversarial Network (GAN), two networks are present and compete with each other on a task. The first network, Generator, is responsible for processing its input and generating new data from the input, pertaining to the task. It starts with a random distribution, and tries to replicate the distribution followed by the input-output pairs of data. The second network, Discriminator, takes the output of the generator and classifies whether it satisfies the condition of the task or not. Taking the example of generating images of dogs from images of cats, the generator processes images of cats and tries to create a realistic image of a dog from it. The discriminator then classifies whether the output of the generator is that of a realistic dog or not. In this sense, both the generator and the discriminator networks compete against each other in a min-max environment of optimization.

While training, the discriminator is provided with a mixed set of outputs generated by the generator network, and ground truth outputs. The ones by the generator are labelled as negatives, and the ground truth is, of course, labelled as positives. The discriminator is then trained on these images. After that, the weights of the discriminator are frozen and the generator's weights are modified based on the feedback of the discriminator. This process is repeated over and over until the losses of both the generator and the discriminator reach a standstill. This is why the optimization problem of the GAN as a whole is termed as an evolving loss landscape. We are not emphasizing minimum discriminator loss, or minimum generator loss. We are, rather, looking for an equilibrium between these two networks and their losses. With respect to images and image generation, the discriminator loss is called as the content loss, while the generator loss (or the whole GAN's loss, technically speaking) is termed as the adversarial loss.

With respect to super resolution, the generator network tries to create a HR image from an LR image. SR-GAN [4] (Super Resolution – GAN) is considered to be a significant work in the super resolution space. One of the more important contributions made through SRGAN is that the content loss, conventionally Mean Squared Error loss (MSE), is replaced with VGG loss wherein the output feature vectors when the images are passed through a VGG network are compared. This was chosen as this loss was more invariant to pixel changes. One low-resolution image may have many variations of high-resolution images from different angles and textures. This pixel-change invariance in the VGG loss helps overcome this many-to-one function problem. The metric used to score this SRGAN is mean opinion score.

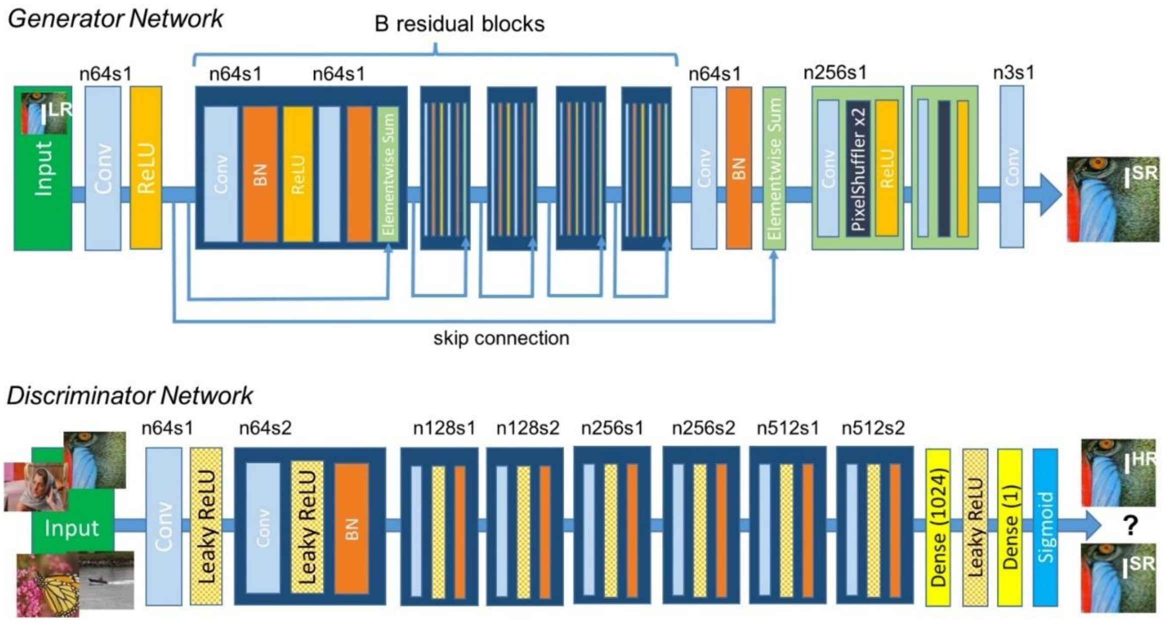


Figure 2.6: SRGAN Architecture [4]

Generator:

The main part of the generator is the recursive usage of multiple residual blocks. A residual block is a block of layers, conventionally a mix of convolutional layers, activation layers and a batch normalization layer, wherein there exist skip connections from the input to the output.

Residual blocks, introduced alongside the popular ResNet architecture, were invented to solve the diminishing gradients problem. Adding more layers to the network, in the days before residual blocks, could prove to be detrimental. This is because if the loss function determines that the contribution made by the added layer is not significant, the weight decay employed by the back propagation algorithm will make the activations of that particular layer to be closer to zero. This affects the training of the earlier layers of the network that come before this particular layer. But if skip connections are introduced, the activation of the previous layers would atleast be present even if the training process diminishes the gradients of the deeper layers. Thus, information is preserved and is transported back to the earlier layers for weight modification. This way, deeper networks can be formed without fear of the diminishing gradients problem.

Once the image is passed through the 16 residual blocks, it is passed through another convolutional layer with 256 filters in it. This means that the output of this convolutional layer has 256 channels in it. This then goes through a special layer known as a pixel shuffler. What this layer does is that it shuffles the pixels on the various channels in such a way that the height and width of the output is multiplied by a certain quantity.

For example, if the output of the previous layer is (C*16)*H*W, where C – Channels, H – Height, and W – Width, the pixel shuffler can transform this to C*(H*4)*(W*4), effectively increasing the resolution of the image. This is found to be a very effective method of upscaling an image via deep learning. After this, the output is passed through a convolutional layer of 3 filters, giving back the 3 RGB channels to the output and the normal dimensions of an image.

Discriminator:

The discriminator is a conventional deep neural network with multiple blocks of convolutional, activation and normalization layers. The convolutional part is then followed by some dense layers full of neurons, leading to the decision in the end. The decision for the network to make in this case is if the image given as input is a real image or a super-resolved image.

2.4.2 Enhanced SRGAN (ESRGAN)

Enhanced SRGAN [7], or ESRGAN, is a network developed on top of SRGAN and is mainly meant for super resolution of single images (SISR). The network is much deeper in ESRGAN than it is in SRGAN, and this leads to better feature extraction. It uses a Residual in Residual Dense Block (RRDB) instead of the Residual Blocks used in SRGAN. These are deeper and complex blocks than the SRGAN, in that they involve more layers in a single block. But they still maintain the residual nature of the network so that they can avoid the diminishing gradients problem.

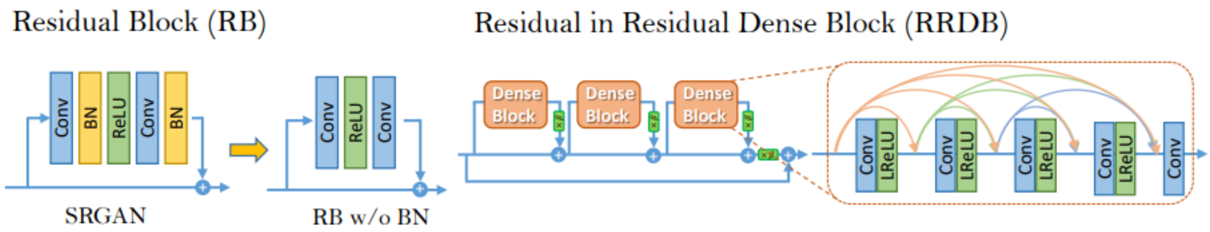


Figure 2.7: Residual in Residual Dense Block [7]

This is implemented in the generator network. As you can see above, an RRDB block is composed of 3 dense blocks, which again is made of 4 conv-layers and leaky ReLU activation layer pairs and one convolutional layer in the end. And as such, 23 such RRDB blocks are used in the generator, giving the generator more room to extract much more intricate features. As you can see in figure 2.8, the features are much sharper with ESRGAN than in SRGAN.

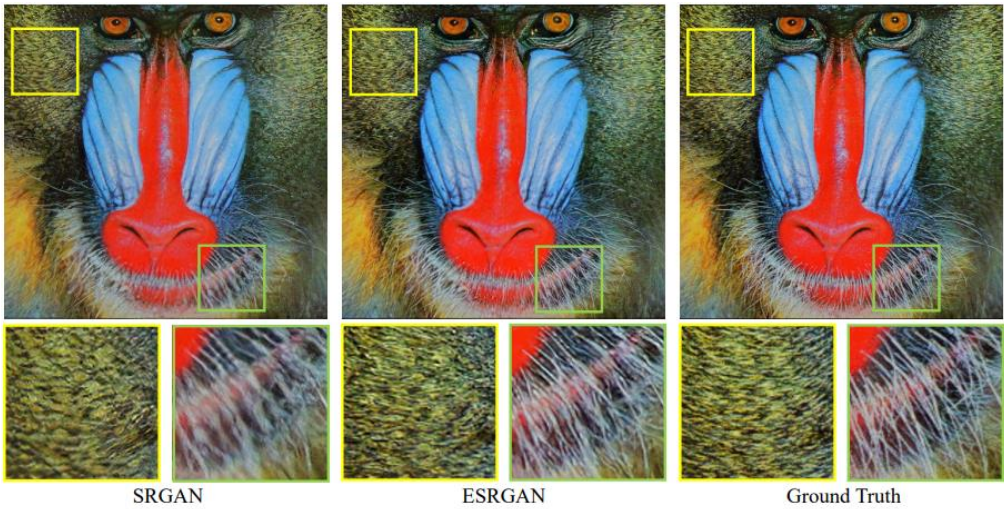


Figure 2.8: ESRGAN output comparison [7]

2.5 Software/Packages Used

2.5.1 Python programming language

We primarily use python as the language of choice to implement our project. This decision is influenced by the fact the Python is currently the most popular tool of choice for machine learning, and has excellent support from the community for the same. Important deep learning packages like Tensorflow and Pytorch are available on python, and are easy to access and use. Research and experimentation become much faster with the easy-to-use characteristic of python and the packages available in python. Recent research in this field is almost always implemented in python, further solidifying our choice.

2.5.2 Tensorflow, Pytorch and Keras

Tensorflow and Pytorch are two of the most popular open-source Deep Learning packages/libraries available in Python. They are used for constructing and implementing neural networks and other machine learning algorithms.

Developed by the Google Brain team, Tensorflow also provides a wrapper called Keras, which abstracts most of the code needed to construct and run algorithms and makes the code for such algorithms easy to write and understandable. Specifically, Tensorflow 2.0 is being used in this project.

Pytorch was developed by the Facebook AI Research Lab, and is a direct competitor to Tensorflow. It is considered to be more popular in the research community than Tensorflow because of the customizability it offers in the construction of neural networks and other algorithms, and the control over the inside processes.

2.5.3 OpenCV

OpenCV is an open-source library that is primarily for Computer Vision purposes, and contains a host of functions for the same. It was originally coded in C++, but a python API (Application Programming Interface) is available and is popularly used by the community.
It provides a lot of functions and tools for image processing purposes, and is highly optimized for such tasks. They also provide optimizations for tasks that are run on CPU, GPU and on edge.

It is cross-platform, and also has bindings in Java, and is supported by Windows, Linux, MacOS, iOS, and Android.

2.6 Summary

In this section, we discussed the theory behind object detection, super resolution, SRGAN and ESRGAN. We also discussed the software packages predominantly used in this project.

CHAPTER 3
DRONE DETECTION AND TRACKING WITH SUPER RESOLUTION IMPLEMENTATION

In this chapter, I give a detailed explanation of the various phases of the project that I worked on before reaching the end goal.

3.1 Phase-1: Super Resolution as a technique to improve drone detection results

In phase one, the possibility of super resolution being used to improve drone detection results was analysed. For this purpose, ESRGAN was chosen as the super resolution algorithm of choice. Various different drone images were collected from the web and from popular datasets such as the USC drone dataset for testing purposes.

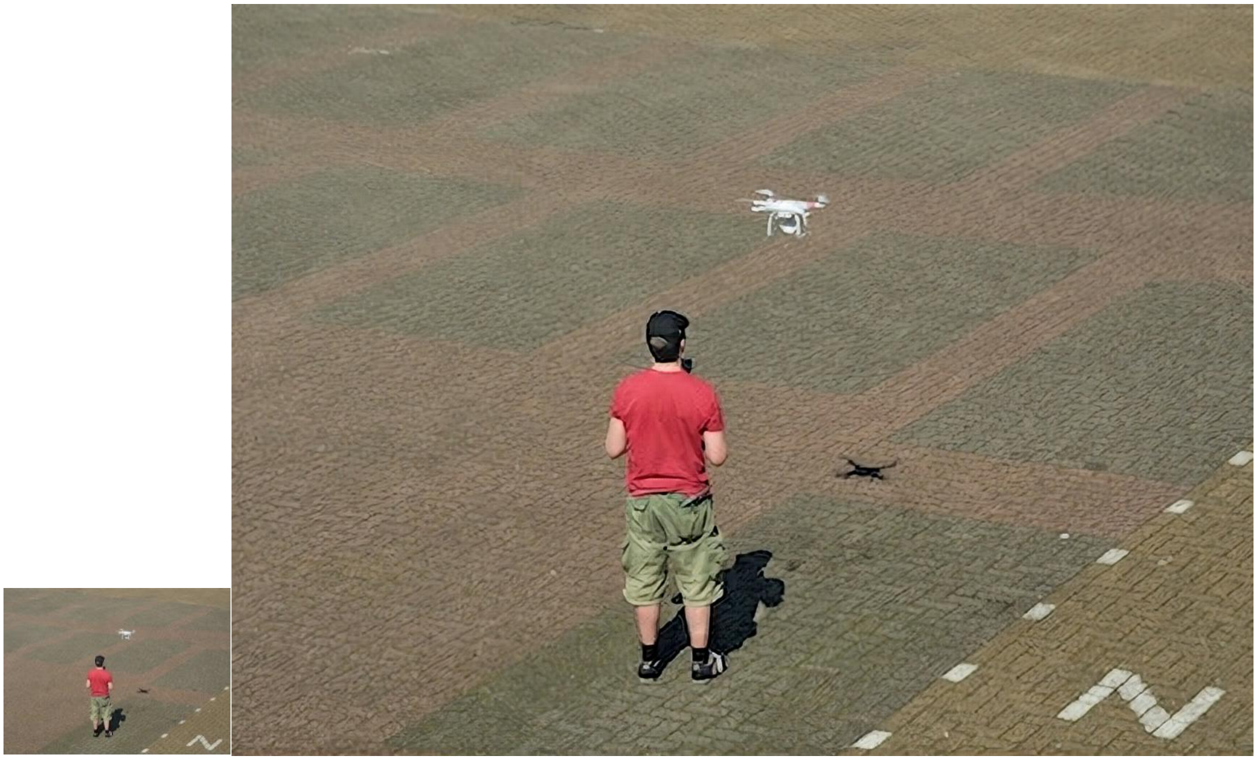


Figure 3.1: ESRGAN sample

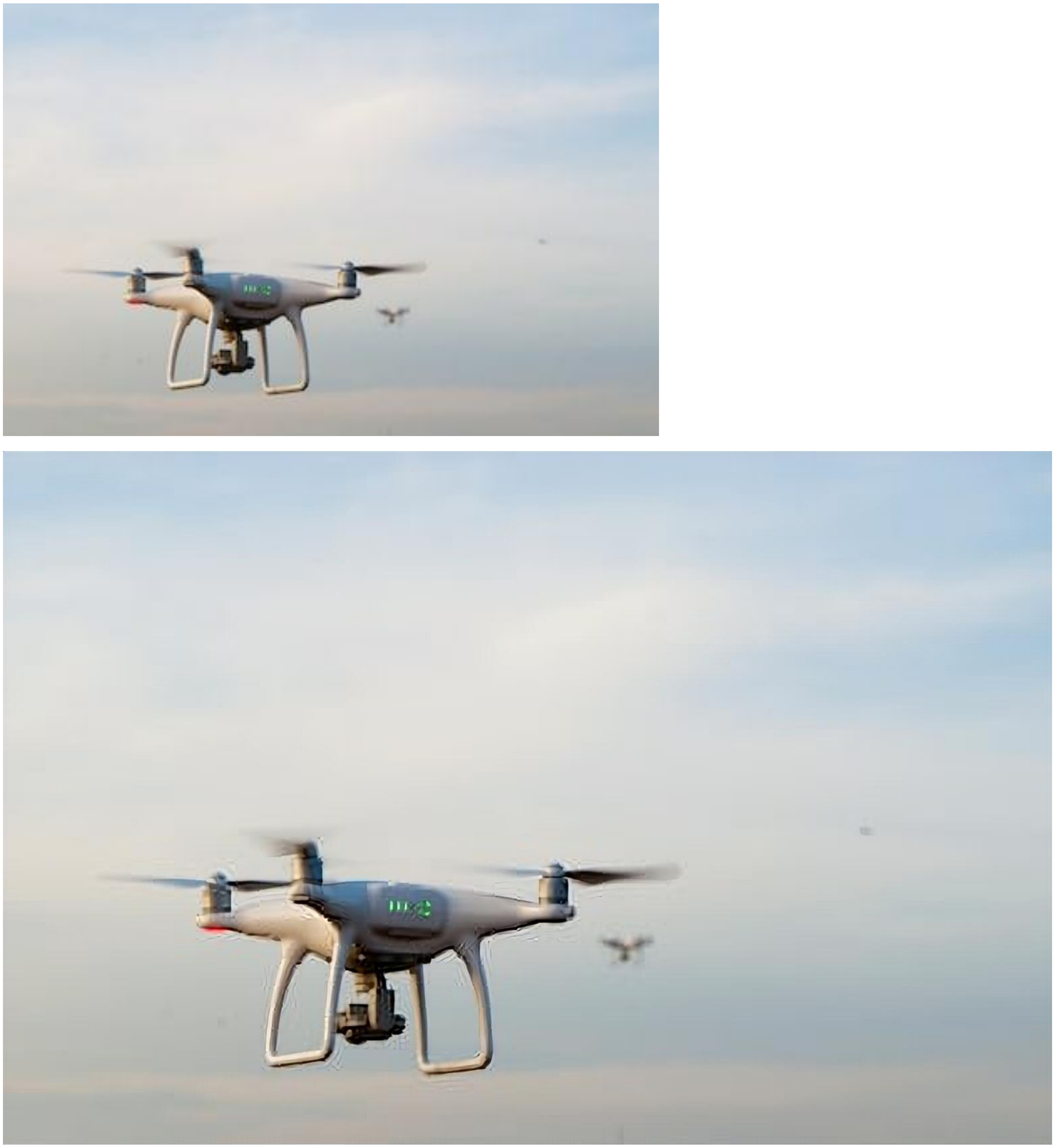
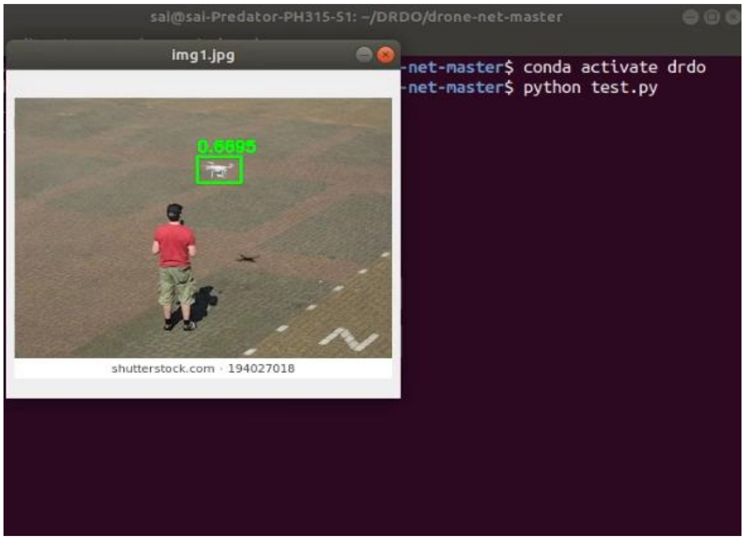


Figure 3.2: Enhanced images of drones

As can be seen from the samples shown (figures 3.1, 3.2), the ESRGAN algorithm works suitably well enough for the task at hand. The next task is to implement object detection with super resolution as part of the pipeline.

For the detector, a YOLOv4 pre-trained on DJI drone images dataset was chosen. As for the test data, we used images from the USC-MCL drone dataset. For proof of concept, the whole image was enhanced using the ESRGAN to increase its resolution by 4 times. First off, without any super resolution, the detector was applied on the image. Even though the drone was detected, the accuracy was noted to be around 66-67%.


sai@sai-Predator-PH315-51: ~/DRDO/drone-net-master
img1.jpg
0.6695
shutterstock.com · 194027018

```
-net-master$ conda activate drdo
-net-master$ python test.py
```

Figure 3.3: Detection without SR

Then, the same detector was used, but after the image was enhanced by ESRGAN. The image was resized for visualization purposes.

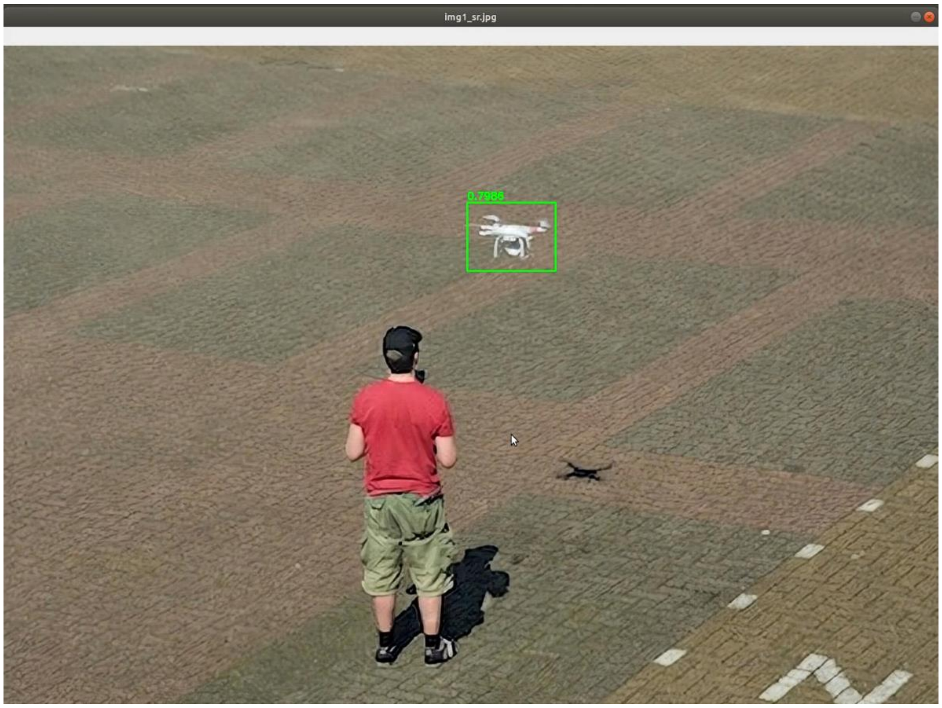
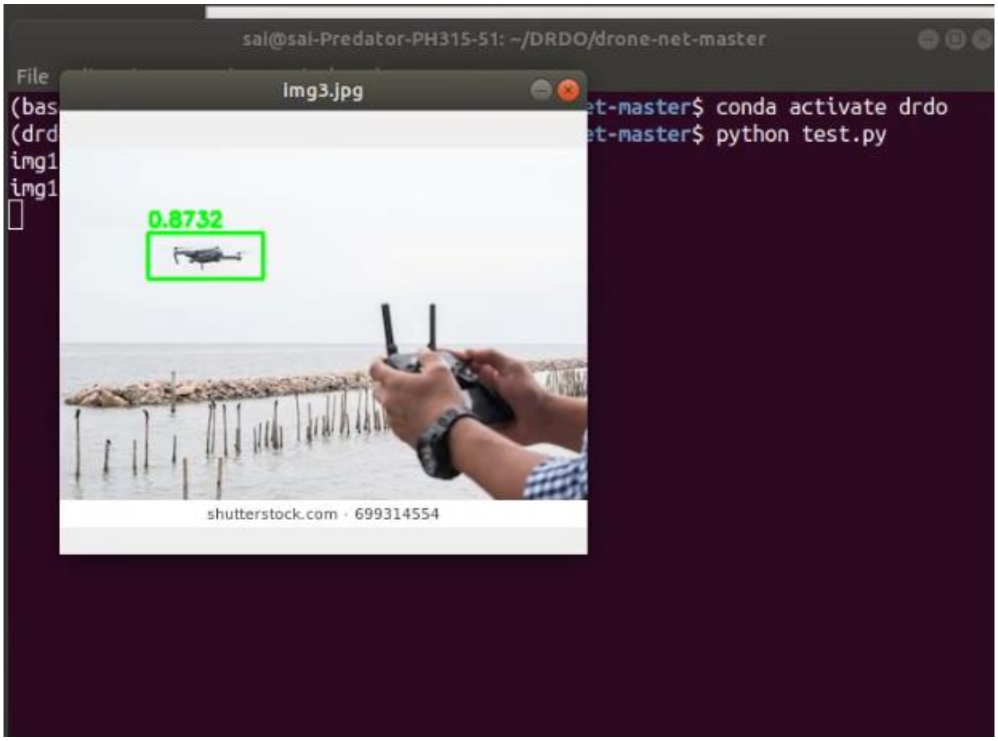


Figure 3.4: Detection with SR on whole image

This time, the accuracy jumped to almost 80%. This is quite a significant jump in detection accuracy. In another example,



```
sal@sal-Predator-PH315-51: ~/DRDO/drone-net-master
File
(bas                                                  et-master$ conda activate drdo
(drd                                                  et-master$ python test.py
img1
img1
```

Figure 3.5: Detection without SR sample 2

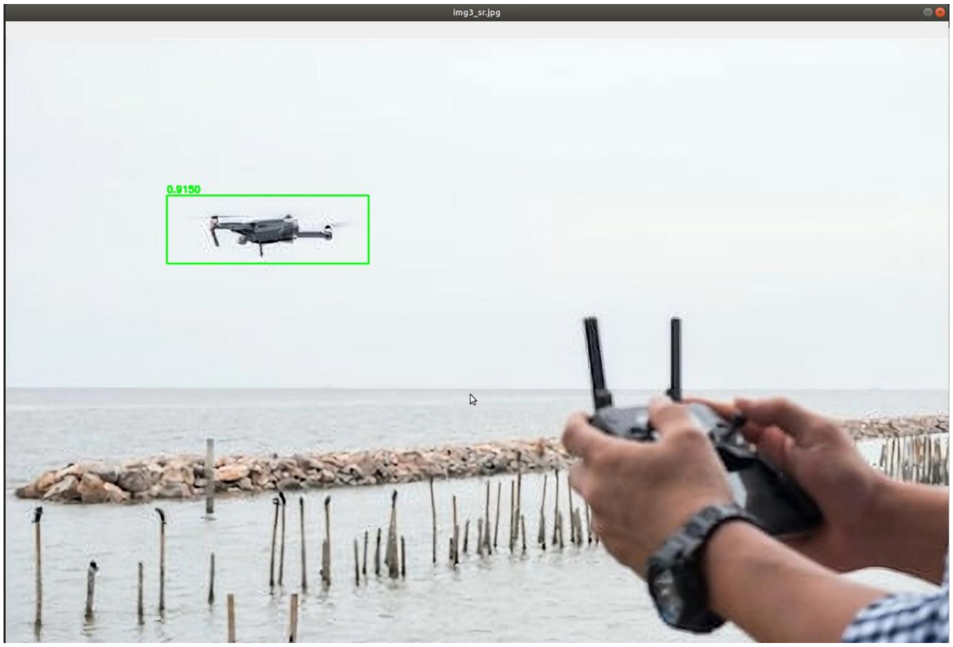


Figure 3.6: Detection with SR sample 2

Here too, we can see that there is a bump in accuracy after super resolution was included in the pipeline. This serves well as a proof of concept that super resolution can cause some significant improvement in detection accuracies.

3.2 Phase-2: Tile-wise detection and merging, region proposal for detection

In phase 2, we proceeded with improving our drone detection using super resolution algorithm. First, we sought to apply some form of region proposal mechanism to automate the process of drone detection and make it easier. Among many popular region proposal algorithms, we chose to proceed with **Selective Search** algorithm for region proposal.

The selective search algorithm by Uijlings et al. performs segmentation of the image by superpixel algorithm, and merges pixels which are similar across several criterion, such as: Shape, color, texture, size, etc. The merged pixels are therefore considered to be areas of interest, wherein individual objects may lie. Bounding boxes are placed over these regions of interest (ROI). This way, the individual regions can be given emphasis during object recognition.

In our case, we experimented with the selective search algorithm on drone images. Our plan was to use ESRGAN to enhance the individual ROIs and then apply the detection algorithm on the enhanced ROIs. But this expectation was not met.

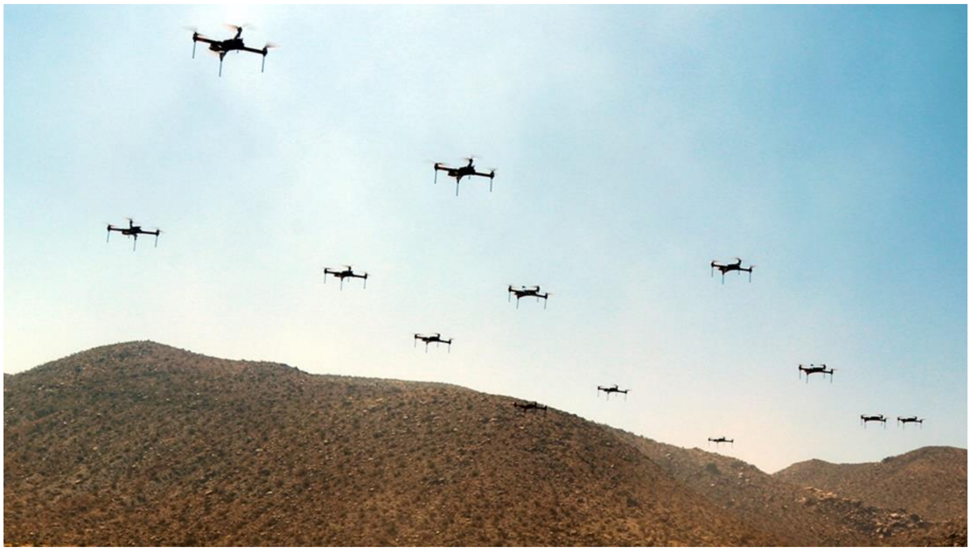


Figure 3.7: Image of multiple drones in the sky

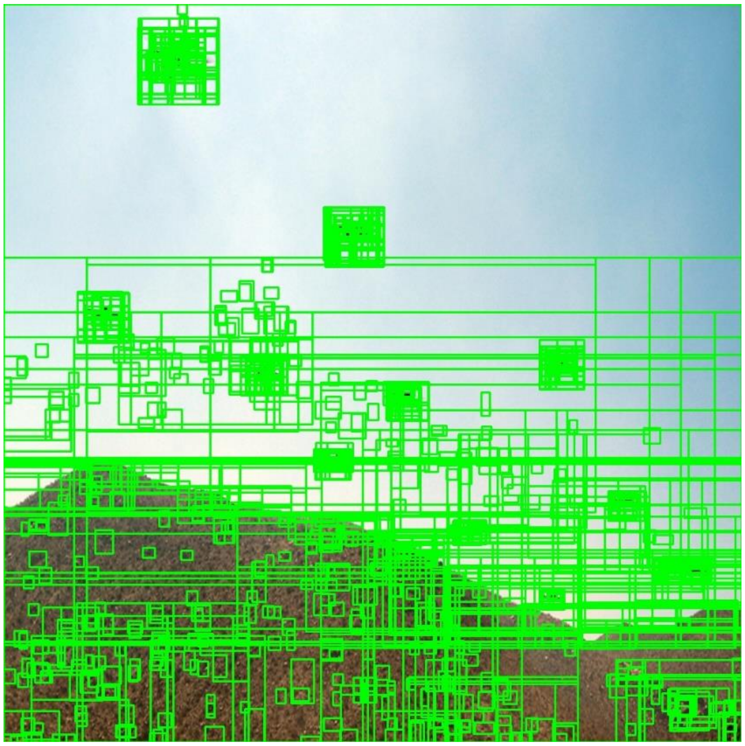


Figure 3.8: Selective Search in action

As shown in figure 3.8, the selective search algorithm resulted in too many ROIs in the image. In the above image alone, it made over ~800 ROIs. This made it rather impractical to use a detector on all the displayed ROIs.

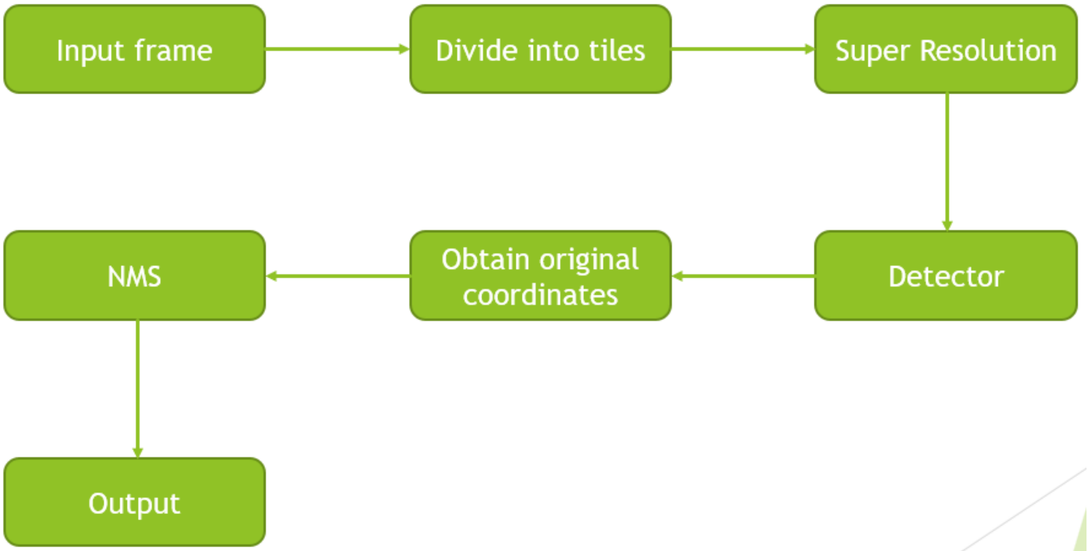


Figure 3.9: Process flow for tile-division algorithm

Instead, we went ahead with a different approach. We sought to divide the image into square tiles, use the super resolution algorithm on each of those tiles and make detections individually, before merging all the detections. The input size for the YOLO algorithm is 416x416, so we went ahead with a tile size of 104x104. After super resolution, the size of tile, or patch, would be increased by 4 times to 416x416. We tested out the algorithm in two ways: Dividing the image into 16 tiles, and then into 100 tiles.

For the 16-tile division, the image was resized to 416x416 to suit the method. Let's take the example shown in figure 3.10. It is a shot taken from a clip in the USC drone dataset.

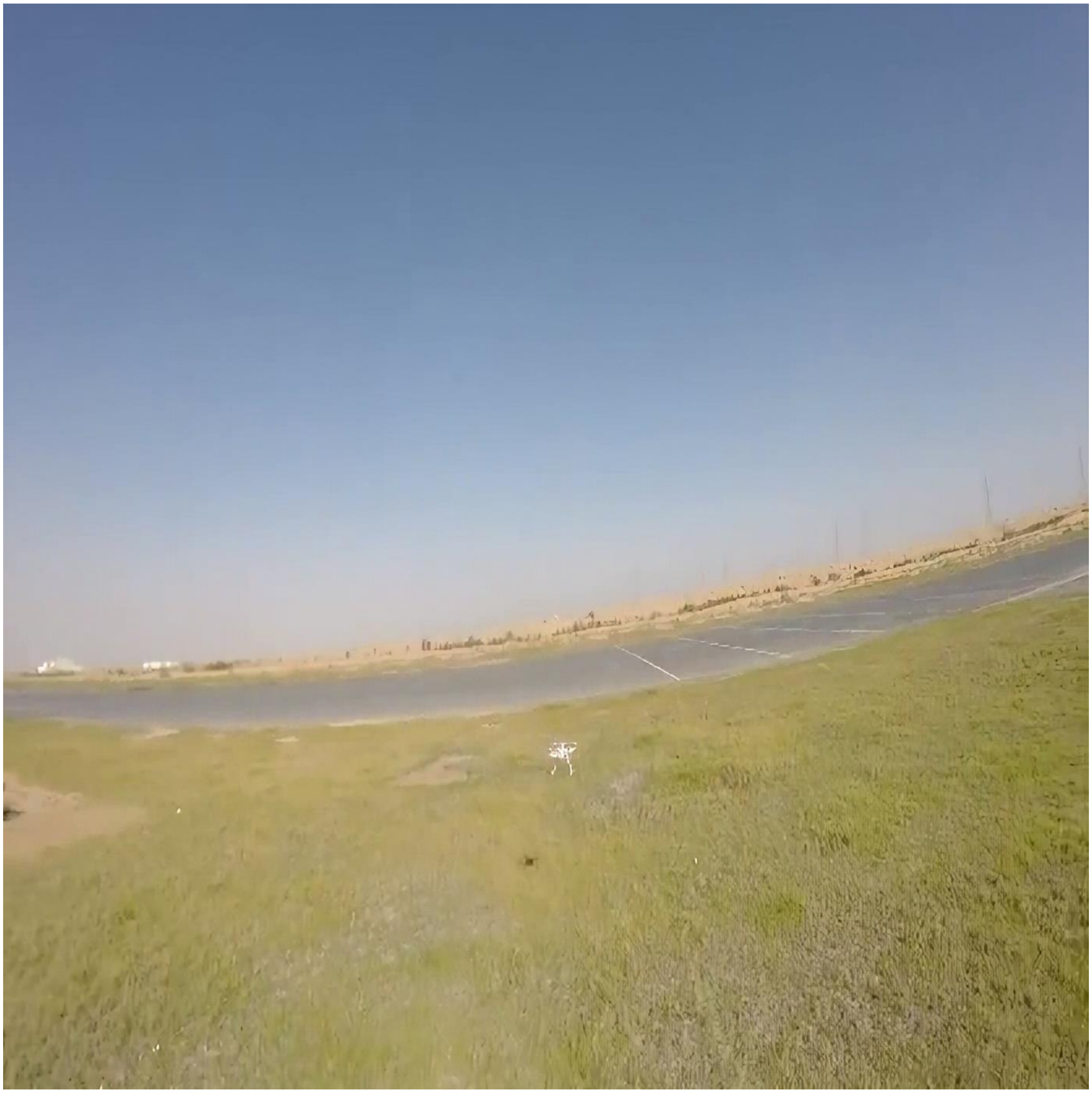


Figure 3.10: Sample image from USC drone dataset [1]

You can see that the size of the drone in this picture is incredibly small compared to the size of the picture itself, and thus it serves as an ideal test subject. Both direct object detection, and detection after enhancing the whole image with super resolution, failed to produce any results. Proceeding to dividing the image into 16 blocks, we first resize the entire image to 416x416. Next, we divide the image into 16 patches, with each patch of size 104x104. Passing these patches individually through ESRGAN, we get 416x416 sized patches. This is then passed through the detection algorithm and detections are made. After which, we obtain the relative coordinates for the detections, with respect to the unenhanced 104x104 sized patches. Then, these bounding boxes from the individual patches are merged for the whole image.

While dividing the image, there is a good chance that multiple patches share the same object. This is where **Non-Maximum Suppression (NMS)** is used.

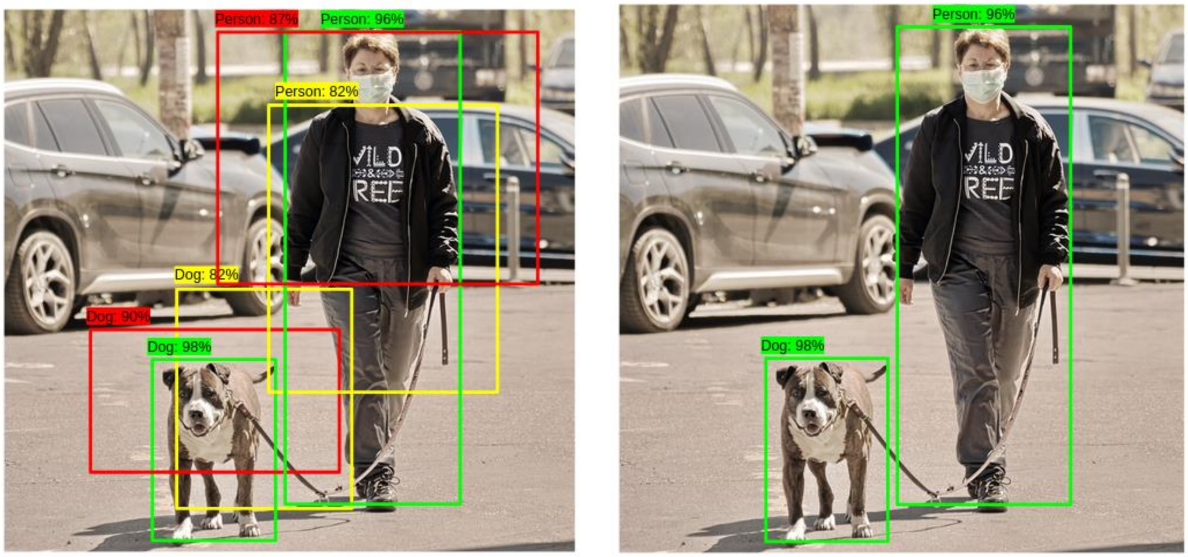


Figure 3.11: Non-Maximum Suppression

For example, in figure 3.11, you can see that while there are only two objects in the image that are significant, multiple detections are made for them. Non-Max Suppression algorithm keeps only the bounding box with maximum detection accuracy and discards all the other bounding boxes, for each object detected in the picture.

The same process happens for 1040x1040, 100-tile division as well. Only in that case, the image is resized to 1040x1040. The images that we were dealing with were thousands of pixels in width and height. Our expectation as such was that the 100-tile division would perform better than the 16-tile division, because the image is not resized to so small a resolution that we lose too many pixels as with the 16-tile division. This was a reasonable argument to make, but the results proved to be somewhat otherwise.

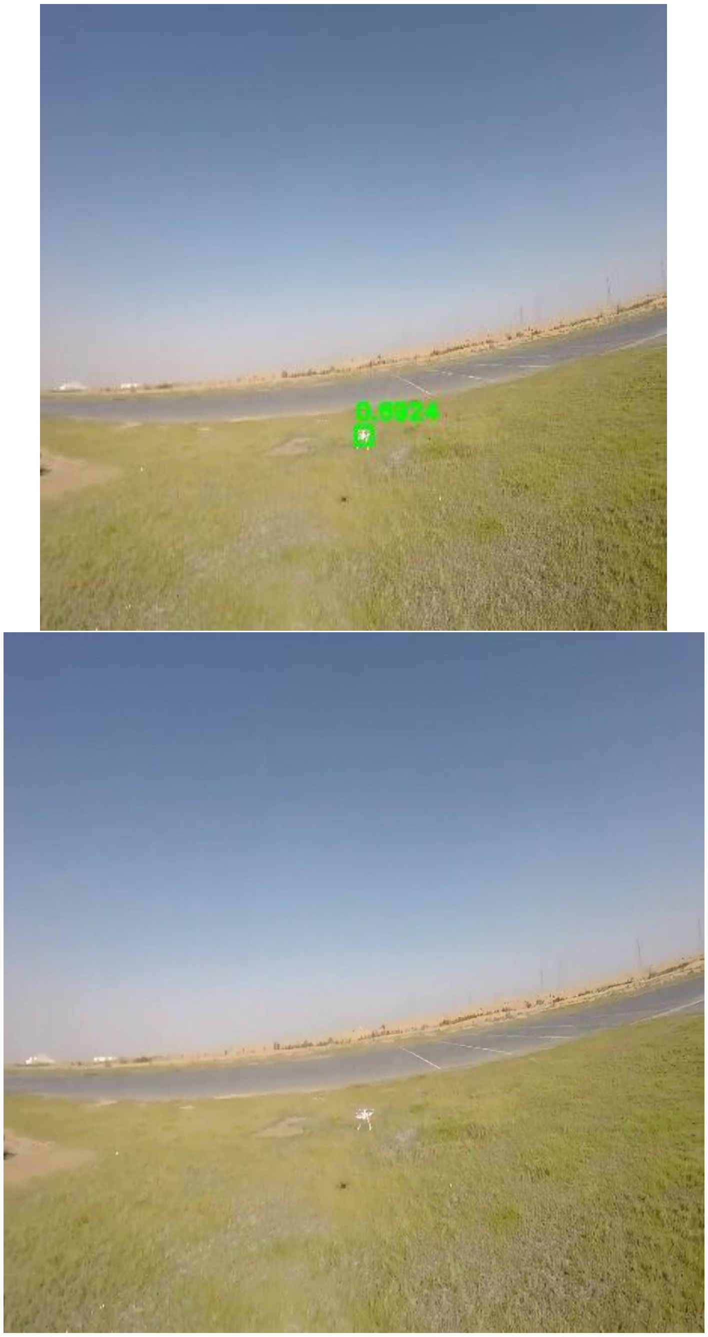


Figure 3.12: 16-tile division (top); 100-tile division (bottom)

We observed that the results were mixed, with the 16-tile division performing better than the 100-tile division in many cases. And in other cases, the 100-tile division appeared to be the best. Here are some more examples to illustrate this.

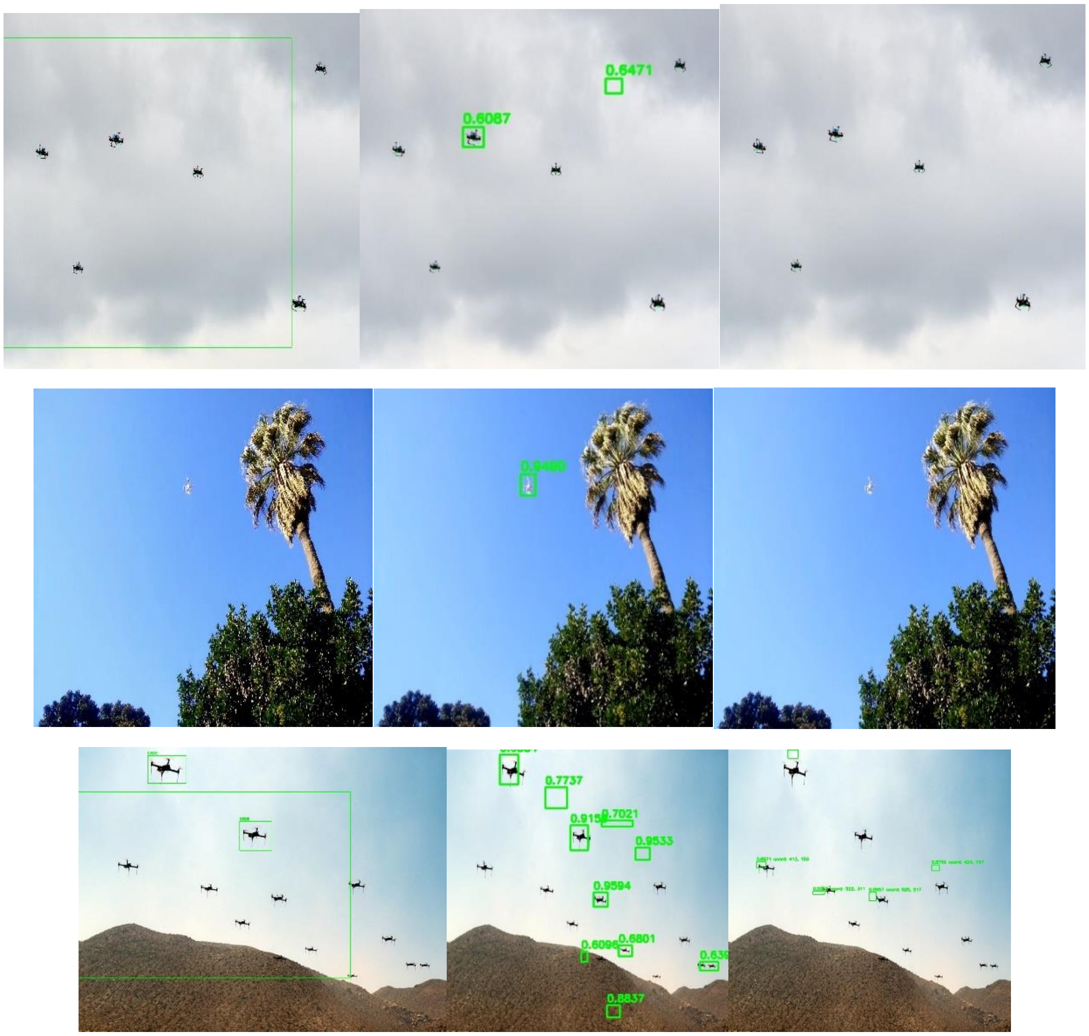


Figure 3.13: Montage of detection outputs with SR on whole image (left); 16-tile division (middle); 100-tile division (right)

As you can see in figure 3.13, there are quite the few mixed results. But in the case of a single drone in an image, especially in the USC drone dataset, the 100-tile division algorithm seems to perform poorly. In other cases, the 100-tile division is observed to make more coherent and relevant detections than the 16-tile division algorithm.

But before we can conclude anything, there is yet another important factor to take into account. Since this effort is directed towards making drone tracking easier, it is mandatory to look at the execution time of both the 16-tile and 100-tile division techniques.

It was observed that the 16-tile division algorithm, intuitively, takes much lesser time than the 100-tile division algorithm. On average, the 16-tile division algorithm takes 10-11s to complete for one image. The 100-tile division algorithm however, takes 60-63s in comparison. These measurements were between the reading of the image (not including) and the displaying of the resulting image (not including). It is also important to note the specifications of the testbench. These were tested on a laptop consisting of a Intel core i5 8th Gen CPU and an Nvidia 1050ti GPU. It was also taken on the Linux Ubuntu-18.04 platform, and care was taken so as to not have any other unnecessary application running in the background as much as possible. It is also important to note that these algorithms are raw, and not optimized for speed. Considering all this information, it is theorized that the 16-tile division algorithm may perhaps be practically usable for tracking scenarios, more so than the 100-tile division algorithm.

3.3 Phase-3: Implementation of drone tracking assisted by Super Resolution

In this phase, we try to implement the 16-tile division algorithm on a correlation tracker. The adaptive correlation filter-based tracker [2] is an improvement on the MOSSE tracker which was based on adaptive correlation filters for translation. Both are detection-based tracking algorithms.

With correlation filter-based tracking, a correlation filter is trained on-line on the detections made by the detection algorithm in such a way that the filter responds to the target in the image with sharp peaks and responds negatively to the background in the frequency domain. This tracking approach is known primarily for its computational efficiency, due to the fact that it relies on fourier transform for computations. As such, it was still uncommon to use correlation trackers because it was sensitive to changes in the target’s appearance. This was improved in the MOSSE tracker, which was better adapted to deal with variations in the appearance of the target and the background. The appearance of the target was modelled by adaptive correlation filters, and the tracking is performed with convolutions.

While the MOSSE tracker worked for objects that translate in any direction, it failed to incorporate change in size of the object as it moves across time. Thus, even if the object increases or decreases in scale according to its distance to the camera, the bounding box around the target remains the same size. But the adaptive correlation filter-based tracker learns separate discriminative correlation filters, based on a scale pyramid representation, for both translation and change in scale. This is a simple motion-based tracker by today’s standards, but is a good starting line for testing our algorithm.

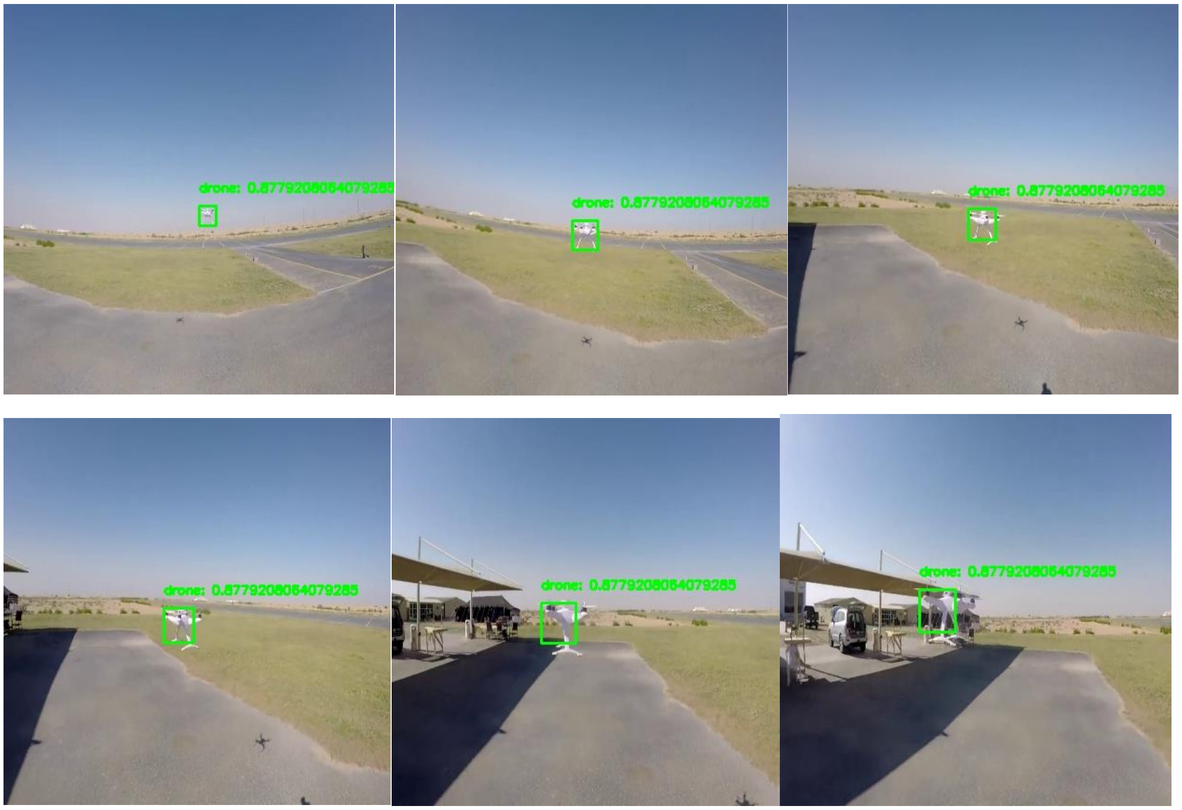

drone: 0.877920806407928
drone: 0.8779208064079285
drone: 0.8779208064079285
drone: 0.8779208064079285
drone: 0.8779208064079285
drone: 0.8779208064079285

Figure 3.14: Detection based tracking using correlation tracker

The above sequence of images illustrates the results of our work. You can see that the detection was made with a good accuracy of 87.7%. Without the super resolution algorithm, it was found that the detection accuracy is around 50%. We should remember that the threshold for detection was also set at 50%, and is common practice to do so. Thus, there is a good chance that the detector alone would fail to notice the same drone in the next frame. And this further goes to show that the detector benefits much from the super resolution algorithm in this case.

3.4 Summary

In phase 1, we strived to create a proof of concept that Super Resolution can impact detection results in drone detection positively. We used super resolution for the whole image, and compared the detection results with that when no super resolution was used on the input image. We succeeded in establishing the proof of concept.

Phase 2 involved experimenting with ways in which super resolution and detection can be applied to an image. We tried the Selective Search algorithm for region proposal, and then resorted to a tile-division technique for super resolution and detection. We experimented with 16-tile division and 100-tile division and compared the results in both time and precision.

In phase 3, we implemented the 16-tile division algorithm on a correlation tracker successfully, and compared results with this technique and without any super resolution.

CHAPTER 4
RESULTS AND DISCUSSIONS

We observed that the 16-tile division algorithm took a comparatively short period of time, around 10-11s, to process each frame and also provides a good amount of boost to the detection results. It must also be noted that this algorithm would take a much shorter time to run if optimized and run on a machine with more compute power. By today’s standards, the Nvidia 1050ti GPU can be considered old.

The 100-tile division algorithm, however, took almost a minute to process each frame. This is impractical to be of use in realistic scenarios. Super resolution on the whole image without any form of resizing is also impractical considering the size of the frames that are being used.

We then went for using a adaptive correlation filter-based tracking algorithm to track the drone across a video, using the 16-tile division and detection algorithm for initial detections. Testing on the USC-MCL dataset, the tracking is observed to be performed well at around 87.7% accuracy in the initial detections.

The aim of this project which was to assess the impact of Super Resolution on drone detection and tracking was fulfilled.

CHAPTER 5
CONCLUSION AND FUTURE WORK

6.1 Conclusion

Hence, we were successfully able to observe the effect of super resolution algorithms on the task of drone detection. The same was also implemented on a simple object tracker and the results were observed to be a success.

6.2 Future work

Throughout the duration of the project, multiple points of improvement could be observed and were noted:

- The input size bottle neck of the YOLOv4 algorithm can be resolved by including a few extra layers before the network that can process input images of preferred size and convert them into feature maps of the input size of YOLOv4. This way, data is not lost by simple resizing, and is preserved in the feature maps. This could be one viable solution to experiment with.
- YOLOv4, while being an exemplary object detection algorithm, is succeeded by various other object detection algorithms that are much faster and more accurate. One such example would be the recent YOLOv5 which has achieved state-of-the-art results in object detection.
- Another point to be noted is that the detector could be trained on more quality data. While images of DJI drones are good to train a drone detector, it is not enough to train a detector that can identify drones flying in the air at a small size/scale. But this is also one of the points where super resolution comes in handy.
- One rather obvious point already conveyed is that the algorithms can be tested with a testbench of more compute power. The capabilities of the testbench system matter quite a bit in the processing time of the algorithms, as can be expected.
- Yet another future endeavour would be to consider optimizing the network to make it run faster. One common way to do it would be to quantize the weights. Frameworks like TFLite and Tensorflow RT provide methods to quantize the weights of the detection network to make it run much faster while maintaining similar results.

REFERENCES

[1] Wang, Ye, Yueru Chen, Jongmoo Choi, and C-C. Jay Kuo. “Towards Visible and Thermal Drone Monitoring with Convolutional Neural Networks.” *APSIPA Transactions on Signal and Information Processing* 8 (2019).

[2] Danelljan, M., Häger, G., Shahbaz Khan, F., & Felsberg, M. (n.d.). *Accurate Scale Estimation for Robust Visual Tracking*.

[3] Uijlings, J. R. R., van de Sande, K. E. A., Gevers, T., & Smeulders, A. W. M. (n.d.). *Selective Search for Object Recognition*. Retrieved April 22, 2022, from http://disi.unitn.it/

[4] Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., & Shi, W. (2016). *Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network*. http://arxiv.org/abs/1609.04802

[5] Chen, Y., Aggarwal, P., Choi, J., & Jay, C. C. (2018). A deep learning approach to drone monitoring. *Proceedings - 9th Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2017*, *2018-February*, 686–691. https://doi.org/10.1109/APSIPA.2017.8282120

[6] Haris, M., Shakhnarovich, G., & Ukita, N. (2021). Task-Driven Super Resolution: Object Detection in Low-Resolution Images. *Communications in Computer and Information Science*, *1516 CCIS*, 387–395. https://doi.org/10.1007/978-3-030-92307-5_45

[7] Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Loy, C. C., Qiao, Y., & Tang, X. (n.d.). *ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks*. Retrieved February 15, 2022, from https://github.com/xinntao/ESRGAN.

[8] Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). *YOLOv4: Optimal Speed and Accuracy of Object Detection*. https://arxiv.org/abs/2004.10934v1

[9] Coluccia, A., Fascista, A., Schumann, A., Sommer, L., Dimou, A., Zarpalas, D., Méndez, M., de la Iglesia, D., González, I., Mercier, J. P., Gagné, G., Mitra, A., & Rajashekar, S. (2021). Drone vs. Bird detection: Deep learning algorithms and results from a grand challenge. *Sensors*, *21*(8). https://doi.org/10.3390/s21082824

[10] Wu, M., Xie, W., Shi, X., Shao, P., & Shi, Z. (2018). Real-time drone detection using deep learning approach. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, *251*, 22–32. https://doi.org/10.1007/978-3-030-00557-3_3

[11] Nalamati, M., Kapoor, A., Saqib, M., Sharma, N., & Blumenstein, M. (n.d.). *Drone Detection in Long-range Surveillance Videos*.

[12] Shermeyer, J., & van Etten, A. (n.d.). *The Effects of Super-Resolution on Object Detection Performance in Satellite Imagery*.

[13] Bashir, S. M. A., Wang, Y., Khan, M., & Niu, Y. (2021). A Comprehensive Review of Deep Learning-based Single Image Super-resolution. *PeerJ Computer Science*, *7*, 1–56. https://doi.org/10.7717/peerj-cs.621

[14] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). *You Only Look Once: Unified, Real-Time Object Detection*. http://arxiv.org/abs/1506.02640

BIODATA

Name:	Sai Krishna M
Phone Number:	+91 63824 26781
Email ID:	sai.krishna2018@vitstudent.ac.in
Permanent Address:	Plot no. 2, Mullai Street, Rambakshi Nagar, Jain Nagar Extn., Chromepet, Chennai - 600044