Robust Object Recognition and Tracking with Drones

Jiaying Wu
Abstract

The Skara Skyddsängel project explores an innovative method of providing illumination for cyclists along a 20km unlit bike lane using drones. Current GNSS approach performs generally well but further improvements are need for better robustness. Consequently, this thesis project is raised to seek a robust solution in the field of computer vision. YOLOv7 is one of the most advanced object detection algorithms. Considering that this will be applied to drones, we opted for the compact version, YOLOv7-tiny. To enhance the performance of the algorithm, we created a custom dataset and expanded it through data augmentation techniques. After our comparative experiments, the model reached a favourable mean average precision (mAP) of 98.88%. Evaluation on unseen videos shows that the trained model can effectively meet the specific requirements of this unique scenario.
Acknowledgements

I’d like to extend my gratitude to my supervisor Lei Chen for his helpful instruction throughout the process. Thanks also go to my reviewer Didem Gürdür Broo for her valuable feedback.

I am also grateful to my parents for their ongoing support.

This study is made possible through the support of the Vinnova-financed project, Skara Skyddsängel.
Table of Contents

List of Tables

List of Figures

List of Acronyms

1 Introduction 1
   1.1 Motivation .............................................. 1
   1.2 Relevant Studies ........................................ 2
   1.3 Research Questions ...................................... 3
   1.4 Outline .................................................. 4

2 Background 5
   2.1 Image processing ......................................... 5
   2.2 Object Detection .......................................... 6
      2.2.1 Computer Vision ...................................... 6
      2.2.2 Convolutional Neural Networks ....................... 6
      2.2.3 Method of Object Recognition ....................... 7
   2.3 YOLOv7-tiny ............................................. 8
      2.3.1 CBL Module .......................................... 8
      2.3.2 E-ELAN Module ..................................... 10
      2.3.3 MPConv Module .................................... 10
      2.3.4 SPPCSP Module .................................... 11
      2.3.5 Overall Architecture ................................ 11

3 Experiment 14
   3.1 Dataset .................................................. 14
      3.1.1 Video Source ......................................... 14
      3.1.2 Frame Extraction ..................................... 17
      3.1.3 Data Annotation ..................................... 17
      3.1.4 Data Preprocessing .................................. 17
      3.1.5 Data Augmentation .................................. 19
      3.1.6 Dataset Description ................................ 20
   3.2 Simulation Tool .......................................... 21
   3.3 Evaluation Metrics ....................................... 22
      3.3.1 Precision and Recall ................................. 22

© Uppsala University

Jiaying Wu
Table of Contents

3.3.2 Intersection over Union ........................................ 22
3.3.3 Non-Maximum Suppression Algorithm ......................... 23
3.3.4 AP and mAP ................................................... 23
3.3.5 Parameters Setting ............................................. 24
3.3.6 Experimental Configuration .................................... 24

4 Results & Discussion .............................................. 25
4.1 Training Results .................................................. 25
4.1.1 The best result ................................................. 25
4.1.2 Analysis of Previous Experiments ............................. 26
4.2 Testing Results ...................................................... 28
4.2.1 Result from the test dataset .................................. 28
4.2.2 Result from the unseen video ................................. 30
4.3 Discussion ........................................................... 32

5 Conclusion & Future Work .......................................... 34
5.1 Conclusion .......................................................... 34
5.2 Potential Future Improvement .................................... 34

Literature ............................................................... 36
List of Tables

Table 3.1: Training dataset classified by different features. . . . . . . . . . . . . . . 20
Table 3.2: Parameters for model training. . . . . . . . . . . . . . . . . . . . . . . . 24
Table 3.3: Experimental Configuration. . . . . . . . . . . . . . . . . . . . . . . . . 24
List of Figures

Figure 1.1: Drones follow cyclists and provide lighting. .................................. 2
Figure 1.2: Drones lose track of cyclist. .......................................................... 2

Figure 2.1: Example image. ................................................................. 5
Figure 2.2: Zoom in to see pixels individually. ............................................ 5
Figure 2.3: The model of one neuron. ....................................................... 7
Figure 2.4: The structure of CBL module. .................................................. 8
Figure 2.5: The plot of Rectified Linear Unit (ReLU) function and Leaky ReLU
function. ................................................................. 10
Figure 2.6: The structure of E-ELAN module. ........................................... 10
Figure 2.7: The structure of MPConv module. ............................................. 10
Figure 2.8: The structure of SPPCSP module. .............................................. 11
Figure 2.9: The structure of yolov7-tiny model. .......................................... 12
Figure 2.10: Predicted result with bounding box, label and confidence score. . . . 12
Figure 2.11: Gradient descent at different learning rate. ......................... 13

Figure 3.1: Workflow of dataset collection. ............................................... 14
Figure 3.2: One frame from the drone when shooting right above. ............ 15
Figure 3.3: One frame from the drone when shooting from the side. .......... 15
Figure 3.4: Cyclist within good visibility. ............................................... 16
Figure 3.5: Cyclist within not good visibility. .......................................... 16
Figure 3.6: The frame contains several light spot far away. ....................... 18
Figure 3.7: Annotation examples. .......................................................... 18
Figure 3.8: Before cropping. ................................................................. 19
Figure 3.9: After cropping. ................................................................. 19
Figure 3.10: Before adjustment. ............................................................ 20
Figure 3.11: After adjustment. .............................................................. 20
Figure 3.12: Interface of AirSim running a multirotor in the Landscape Mountains
template background. ................................................................. 21
Figure 3.13: The view from the multirotor after switched camera. ............ 21
Figure 3.14: Simple explanation of IoU. .................................................. 23

Figure 4.1: Training loss-best result. ....................................................... 25
Figure 4.2: Validation loss-best result. ..................................................... 25
Figure 4.3: mAP-best result. ................................................................. 26
Figure 4.4: mAP-unshuffled dataset. .......................................................... 27
List of Figures

Figure 4.5: Val loss-unshuffled dataset. ................................................. 27
Figure 4.6: mAP-oversized batch. ......................................................... 28
Figure 4.7: Loss-oversized batch. ......................................................... 28
Figure 4.8: Test image 1. ................................................................. 29
Figure 4.9: Test image 2. ................................................................. 29
Figure 4.10: Test image 3. ............................................................... 29
Figure 4.11: Test image 4. ............................................................... 29
Figure 4.12: Wrong detection 1. ......................................................... 30
Figure 4.13: Wrong detection 2. ......................................................... 30
Figure 4.14: Before crossing the bridge. ............................................ 30
Figure 4.15: After crossing the bridge. .............................................. 31
Figure 4.16: Lighting variation 1. ....................................................... 31
Figure 4.17: Lighting variation 2. ....................................................... 32
List of Acronyms

YOLO  You Only Look Once
GNSS  Global Navigation Satellite System
SVM   Support Vector Machine
OCR   Optical Character Recognition
CNN   Convolutional Neural Network
RPN   Region Proposal Network
R-CNN Regions with CNN features
ReLU  Rectified Linear Unit
E-ELAN Extended Efficient Layer Aggregation Networks
MPConv MaxPooling Convolution
SPP   Spatial Pyramid Pooling
TP    True Positive
FP    False Positive
TN    True Negative
FN    False Negative
IoU   Intersection over Union
AP    Average Precision
mAP   mean Average Precision
1 Introduction

1.1 Motivation

Skara, a municipality located in Västra Götaland, is renowned for its wealth of history and offers a bike-friendly environment which makes it possible to explore the center through cycling. Moreover, its extensive cycling path network with well connection to other region allow more opportunities for longer journeys to picturesque surrounding towns and villages as well.

There is one 20-kilometer-route which predominantly along the path of the former railway embankment of the discontinued Gothenburg-Skara and Skara-Skövde railway lines. Cyclist could reach Ardala, Axvall and Varhem through this. However, it’s not always possible to have good lightning conditions on all cycle path on account of the great expense of installation and maintenance of lighting systems. The whole path would plunge into darkness during evening time, where normal bicycle lights could only brighten a narrow stretch of the road.

To reduce potential safety hazards, an ongoing project are developing drones to provide on-demand lighting services to cyclists, which is expected to be an affordable alternative to expensive street lightning. The solution requires the drones to track the cyclist based on predefined parameters, ensuring appropriate lighting range, strength, and other factors. Being equipped with two LED lights with a total of 32 watts and guided through Global Navigation Satellite System (GNSS) from smartphones, the current approach is working generally as expected as shown in Figure 1.1.
However, new issue is raised when the cyclist makes a turn as Figure 1.2. Due to the instability of the GNSS, the drone would lose track of the cyclist and not provide lighting within the specified range anymore. To ensure the safety and experience of cyclist, efforts should be made to improve the robustness of the solution.

1.2 Relevant Studies

Drones are no longer a novel concept to contemporary society: they are playing an increasingly crucial role in civilian life [1]. In urban areas, they have been employed to collect road traffic information in a more efficient way instead of installing traditional street cameras [2]. In rural area, drones can hover over fields for timely pest and disease
Monitoring [3]. Also, there’re some companies leverage the capability of drones to swiftly deliver medical supplies to areas that are challenging to access via conventional means of transportation [4].

One of the most state-of-the-art object detection algorithms You Only Look Once (YOLO) has been proven to provide significant assistance in solving many real-life issues. Wang et al. applied an improved GF-YOLOv7 network model to carry out bouncing locks detection in substation power cabinets to better meet maintenance requirements [5]. Zhang et al. proposes a lightweight surface target detection algorithm built upon YOLOv7 to support the unmanned surface vehicle while executing missions in more complex circumstances [6]. Furthermore, experiments have been conducted to integrate YOLOv3 into a proposed system, which designed for use during an epidemic, and testing has been carried out using the Gazebo simulation environment [7].

Narrow the scope to studies with similar detection target, Li et al. conducted experiments using a detection method with a linear Support Vector Machine (SVM) classifier and efficient HOG-LP feature extraction. The intention is for installation on a moving vehicle’s vision systems to protect crossing cyclist [8]. The task in [9] also aiming at detection in video streams. Shen et al. used yolov3 algorithm to identify red light violations involving pedestrians and non-motor vehicles. They combined the results from both face detection and non-motor vehicle detection then applied a location score function to filter out false detections. In studies in dark and GNSS-denied environment, Andersen et al. implemented a deep learning object detector to get the location of manholes in relation to the drone [10]. They investigated four versions of YOLOv5 architecture, varying from sizes, and eventually chose the smallest model for further testing due to the priority on detection speed. Dodia and Kumar conducted comparative studies on three versions of YOLO algorithm (YOLO-v3, YOLO-v5, YOLO-v7) by evaluating vehicle detection [11]. The result indicated that the latest version reached the highest accuracy on the open-source video dataset.

Inspired by these application cases, a promising approach emerges: integrating vision-based object recognition and tracking technology, which could improve cyclist identification and tracking capabilities in darkness, into the system.

1.3 Research Questions

This thesis work will conduct research on the state-of-the-art algorithm to answer two questions:

- RQ1: if the YOLO algorithm can be leveraged to develop a drone-suited model based on the analysis of actual videos captured by drones.
• RQ2: how well this model performs in identifying cyclists under poor ambient light conditions.

1.4 Outline

This report is structured as below:

• Chapter 2: Important concepts related to object detection are briefly reviewed, a detailed introduction to YOLO model network is demonstrated.

• Chapter 3: Description of experiment methodology, including simulation environment, dataset and evaluation metrics.

• Chapter 4: A detailed analysis and discussion towards the result.

• Chapter 5: A summary of this thesis work and suggestion of future work.


2 Background

In this chapter, basic concepts related to the thesis work are mentioned for better understanding. First part is the brief introduction about computer vision, then several object recognition algorithms are listed including the one closely related to this project, YOLOv7. After that, several comprehensive charts demonstrate the architecture of the original model.

2.1 Image processing

Break down a digital image, there will be a large quantity of cubes such as when zooming in Figure 2.1, pixel level is shown in Figure 2.2. Pixels are the smallest units which contains information of color and brightness. Therefore, the quality of the image is up to the terminology resolution, which represents the total amount of pixels. For example, an image with the resolution of 1280 x 1280 is more clear and sharp compared to the one with 640 x 640.

![Figure 2.1: Example image.](image1)

![Figure 2.2: Zoom in to see pixels individually.](image2)

As for machine, the input image is actually a matrix of pixels. Each element in the matrix is a pixel represented by a numeric value. In this project, all of the images are presented in RGB color model, which means each images is composed of three arrays corresponding to the three color channels: red, green and blue.
Image processing is a technique that can assist us extract useful information from a digital image and help machine interpret visual data more easily, it is favoured in tasks such as feature extraction. There are already a large number of applications related to human activities involving image processing such as moving-object tracking, remotely sensed scene interpretation and automatic visual inspection system [12]. This project preprocessed the obtained dataset by applying data augmentation which shares similar mechanism with image enhancement — one of the typical phases of Image processing. This will be a detailed introduction in the third chapter.

### 2.2 Object Detection

#### 2.2.1 Computer Vision

In recent years, machine learning has gained significant popularity from researchers in various domains to the public. In a similar manner to the learning process of human’s which relies on the accumulate experiences to make a decision, machine learning algorithm generate models that capture the complex patterns based on the input dataset, then produce increasingly accurate prediction on unseen instance. It encompasses a diverse range of studies such as algorithm complexity and statistics.

Alongside advancements in machine learning, the field of computer vision has experienced accelerated progress, propelled by the availability of large-scale image datasets, which contribute to remarkable breakthrough in numerous practical applications like vehicle automation and robotic delivery. The essence of computer vision lies in making the machine understands and analyzes vital features which extracted from images and videos in the similar way as human. There are several widely implemented tasks like image segmentation, Optical Character Recognition (OCR) and image classification. For this project, the work will be implemented through object detection.

#### 2.2.2 Convolutional Neural Networks

Deep learning has propelled the development process of object detection significantly. By utilizing methods based on Artificial Neural Networks such as Convolutional Neural Network (CNN), it has expanded the boundaries of what was previously achieved in the realm of image processing [13]. CNN is a feed-forward neural network that can efficiently process spatial visual data and extract representative features, which provides significant assistance in accurate prediction of object categories and coordinates of bounding boxes. Therefore, it has made important contributions to the core architecture that forms the core of numerous state-of-art object detection algorithms.

All of these are established on the groundbreaking research by McCulloch and Pitts. In 1943, they introduced a computational model to simulate how human brain process
received informations and proved that these artificial neurons were capable to perform arithmetic operations theoretically [14]. Based on this, Rosenblatt proposed a pioneering concept, perceptron, and a learning algorithm in 1950s [15]. A single perceptron, also can be called node or neuron, is a basic unit of neural network. It takes the output values of each node from the last layer, weigh them separately, sum up and plus extra bias to adjust the output, pass through activation function(which will be introduced later) then output the final result. The working principle can be explained in Equation 2.1 and Figure 2.3

$$y = f\left(\sum_{i=1}^{i=N} x_i w_i + \text{bias}\right)$$ (2.1)

![Figure 2.3: The model of one neuron.](image)

### 2.2.3 Method of Object Recognition

Over the past decades, deep learning-based object detection approaches have gained extensive attentions from researchers. This approach could be classified into two major categories: one-stage approaches and two-stage approaches.

A pioneering two-stage method, known as Regions with CNN features (R-CNN), suggests potential regions which might contain object of interest based on Region Proposal Network (RPN). The CNN model takes them as input, then conducts feature extraction. Once each region is classified, labels would be assigned to bounding boxes around the objects, alongside the output of corresponding coordinates [16].

However, the attainment of highly precise localization is accompanied by a trade-off in terms of efficiency. As for one-stage algorithms, it reduces computational time notably due to the omission of the separate region proposal stage. YOLO stands out as one of the popular ways in this category. It resizes the input image and divides it equally in forms of $S \times S$ grids. One grid is responsible for detecting the object if the center falls within the
range of the it. Then all the information generated in the two-staged method is delivered in one single shot. Moreover, compared to another advanced detection method Fast R-CNN, it avoids misrecognizing the background as the target object through the global prediction [17].

Since the launch of YOLO in 2016, many researchers have continuously updated and optimized the series by experimenting with various network architectures, activation functions etc. [18]. Among all the iterations, the main trend is the pursuit of the best balance between speed and accuracy and a better fit for applications with real-time requirements. One of the key instances have changed is anchor box. Inspired by Faster R-CNN, anchor box mechanism was introduced to predict bounding boxes when the speedy YOLO-v1 faced the challenge of having difficulty detecting small-sized objects and objects with unseen ratios, which sacrificed the accuracy a little bit [19]. This concept suited well in subsequent versions. However, in YOLOX, Ge et al. switched the detector to an anchor-free manner and adopted another technique [20]. As for YOLO-v7, which is released later, is still an anchor-based model.

2.3 YOLOv7-tiny

Given the system is expected to perform accurate detection swiftly in real-time scenarios, as one of the most state-of-the-art methods, YOLOv7 stands out as an ideal choice [21]. In addition to the basic YOLOv7 model, the entire series also represent several variations that vary in terms of efficiency, size and accuracy such as YOLOv7-W6 and YOLOv7-X. Due to the necessity for applications that deployed on drones to be as compact as possible, the tracking module in this thesis project is decided to be built based on the lightweight YOLOv7-tiny model [22]. Generally, this new version can be divided into 2 parts. It is derived from YOLOv5 but incorporates the Neck component into the Head section, which is responsible for prediction according to the extracted feature from Backbone part.

The whole 77-layer network is piled up by different modules and is explained in the following part. The overall architecture is then presented in Section 2.3.5.

2.3.1 CBL Module

Figure 2.4 shows CBL module, it consists of one convolutional layer, one batch normalization layer, and activation function Leaky ReLU.

![Figure 2.4: The structure of CBL module.](image)
Convolutional layer

Convolutional layer functions as taking a kernel to extract image features by doing element-wise multiplication and sum operation with the input data matrix \([23]\). This procedure works like filtering, thus the size of the kernel and sliding stride can influence the size of the output feature map.

Batch Normalization

In the complex neural network, the parameters and distribution of input to each layer can vary a lot, which requires a smaller learning rate and hinders the convergence speed. Ioffe and Sergey presented an innovative mechanism named Batch Normalization. It succeeds in significantly accelerating the training speed and improving training effect \([24]\).

In this model, this data preprocessing technique works in hidden layers, which scales the values of different features within a similar range to ensure the stability of the model performance during training.

Activation function

Within the hidden layers, activation function adds non-linearity of the network, it takes the weighted sum and bias from the former layer, then generates results and passes to next layer. This procedure can be refer to Figure 2.3. Without this, the final output will always hold a linear relationship with the input no matter how many hidden layers exist since the arithmetic operation on input data and matrix is linear, which will hinder the model’s performance in complicated scenario, since most data in real life is non-linear.

Some of the frequently used functions are: Sigmoid, Hyperbolic Tangent , Softmax and ReLU. The activation function in YOLOv7-tiny model, Leaky ReLU Equation 2.2, is an improved version of ReLU function. As Figure 2.5 demonstrates, a slight slope is applied to the negative input, which means the gradient will not disappear as the case in ReLU. Otherwise, neurons will turn to inactivate state and stop updating during the back propagation process. Therefore, this activation function stands out for the low computing cost and it enhances the learning ability of the network \([25]\).

\[
\text{Leaky ReLU} = \begin{cases} 
  x, & \text{if } x > 0 \\
  \text{negative_slope} \cdot x, & \text{otherwise}
\end{cases}
\] (2.2)
2.3.2 E-ELAN Module

Figure 2.6 demonstrates the other module which is also one significant alteration in this version. Instead of increasing the quantity of computational blocks, Wang et al. chose to change the architecture and keep the transition layer unchanged as ELAN [26] and proposed this as the Extended Efficient Layer Aggregation Networks (E-ELAN).

Figure 2.6: The structure of E-ELAN module.

2.3.3 MPConv Module

The MaxPooling Convolution (MPConv) module is composed by one max pooling layer and three CBL modules with different strides and kernels. Figure 2.7 shows general structure.

Figure 2.7: The structure of MPConv module.
2.3.4 SPPCSP Module

The SPPCSP module is optimized based on the Spatial Pyramid Pooling (SPP) module [27], it takes the output from Backbone network, then enhances the extracted features by conducting concatenation with extra convolution modules before delivering to the Head component.

![Figure 2.8: The structure of SPPCSP module.](image)

2.3.5 Overall Architecture

The architecture of YOLOv7-tiny model is depicted as Figure 2.9. Initially, the algorithm resizes each input image to 640 x 640 pixels. If the image is in RGB format, it is presented as a 3D array with three color channels and used as input. Passing through the Backbone and Head sections, feature maps are extracted with sizes of 20 x 20, 40 x 40, 80 x 80, respectively. By doing this, YOLO model is capable of detecting multi-scale object within the image. As for each detection, it predicts class probabilities and applies non-maximum suppression to eliminate redundant bounding boxes, only keeps the most accurate result. In the end, the network outputs a list of images including bounding boxes with corresponding class labels and confidence scores like Figure 2.10.
Figure 2.9: The structure of yolov7-tiny model.

Figure 2.10: Predicted result with bounding box, label and confidence score.
**Loss Function**

Throughout the YOLO training process, the model’s parameters are continuously adjusted to enhance its ability to accurately predict objects in the image. During each iteration, the loss function evaluates how closely the prediction result aligns with the ground truth by calculating errors in predicted bounding boxes (Localization Loss), confidence scores (Confidence Loss) and class probabilities (Classification Loss) [28].

**Gradient Descent Algorithm**

The gradient is a vector that indicates the direction of the steepest increase in the lost function. The parameters’ optimization goal is essentially to minimize loss function, this can be achieved through gradient descent algorithm, which will try to find the optimal set of parameters.

\[
p_{n+1} = p_n - \eta \nabla f(p_n)
\]  

(2.3)

The updating procedure can be expressed in Equation 2.3. This algorithm adjusts the parameter in the opposite direction by using the current parameter p substrates it, while the learning rate \( \eta \) determines the magnitude of update [29]. A smaller learning rate means longer convergence time, and there would be possibility that the optimal point has not been reached at the end of the iterations. On the other hand, a larger step will probably miss the optimal point. This can be visualize in Figure 2.11.

![Figure 2.11: Gradient descent at different learning rate.](image)
3 Experiment

This chapter details the labour-intensive dataset preparation and describes the simulation tool AirSim as well as the project workflow. To facilitate a better understanding of the experimental results presented in the chapter right after, model evaluation metrics are explained through mathematical equations.

3.1 Dataset

The overview of the exhausting dataset collection process is described in Figure 3.1:

![Workflow of dataset collection](image)

Figure 3.1: Workflow of dataset collection.

3.1.1 Video Source

In this project, those video clips used for training were collected via a modified drone, the DJI Enterprise, which has been equipped with a lighting system. There is generally two sorts of videos. In one of the batches, the shooting position of drone was directly above the cyclist. For the other case, another drone was used for shooting from the side while those two tracking drones were providing light to cyclist. Figure 3.2 and Figure 3.3 serve as individual examples.
In order to describe the structure of the dataset more clearly, and since the lighting conditions around the cyclist vary according to the flying angle and distance of the drone, another category of lighting intensity is proposed. However, under some circumstances, the dynamic lighting from drones makes it more complicated to deduce whether the environment is light or dark.
One solution is to classify based on how cyclists are annotated. If cyclists themselves are within good visibility, bounding boxes can be drawn upon them as the Figure 3.4 shows, then this belongs to the "light" category. If the illumination is dim, thus it needs the assistance of other parts closely around cyclists, the bounding box would be located in the area as the Figure 3.5 shows.

Figure 3.4: Cyclist within good visibility.

Figure 3.5: Cyclist within not good visibility.
3.1.2 Frame Extraction

Since the resizing will more or less impact model performance, all of the extracted frames in original dataset have been resized to 640 x 640 pixels before hand with the aid of Python script and OpenCV library. The amount of frames is up to the input video. For example, if the video plays at 25 fps, it means every second contains 25 individual images. For better data management, each image is labeled and indicates the video it is generated from.

3.1.3 Data Annotation

MakeSense is a user-friendly online annotation tool which is quite convenient to use without any advanced installation, it supports various output file in formats like YOLO, Pascal VOC, JSON and XML.

Initially there were two classes being considered: cyclist and non-cyclist. The class non-cyclist contains some other tiny light spot in the distance, like the frame example Figure 3.6. After the discussion, these factors were not taken into consideration, out of the following reasons:

- Ideally, if the drone is at an appropriate height relative to the cyclist, the chance of the tracking drone seeing those lights in the distance is not high. This type of videos are from the third drone which is especially placed for collecting more training data, so it may not be very suitable for the training purposes for tracking drones.

- The dataset containing this label is not large enough for training.

- One of the characteristics of this type of frames is that the cyclist is usually pretty small. To make the model focus on the cyclist, part of the image is cropped off after data preprocessing. This step will eliminate these distant components.

Therefore, the plan remains to focus on the label "cyclist" and try to improve model attention. Representative annotation examples are shown in Figure 3.7.

3.1.4 Data Preprocessing

There were more than 30000 frames extracted from the chosen source videos, and only 1195 images are kept for dataset. Considerable effort is made in the process of data cleaning:

- It is necessary to remove identical or nearly identical parts. There are around 30 frames extracted every second, sometimes the sequential changes are not so obvious, which results in excessive data duplication.

- Images in low resolution or contains blurred object should also be filtered out.
Figure 3.6: The frame contains several light spot far away.

Figure 3.7: Annotation examples.
The whole dataset is split into three parts: training, validation and test. The model goes through the training set first, to learn features of the target. The validation set enhances the model’s ability to deal with more unseen data instead of purely memorising those images which it has learnt in the earlier step. At this stage, metrics to describe the performance of the model, like mean Average Precision (mAP) and F1-score are computed, which are quite essential for further improvement. After the model is trained, test set evaluates it with unseen data.

### 3.1.5 Data Augmentation

Data augmentation is one of the effective ways to expand the dataset based on the existing one when there’s only a limited amount of data can be collected. At the same time, it can help supplement a certain type of data with unclear features. This technique enhances model’s generalization ability and contributes to the goal of improving robustness performance.

When working with visual data, researchers can consider various methods like noise-adding, blur, brightness and contrast adjustment, cropping, mosaic, rotation, resizing and zooming. In this project, only two of these methods are adopted due to the features of the original dataset which contributes 41% extra (486 images).

#### Cropping

Select one specific region, which is usually the part that deserves attention, then drop the rest of the image. In this case, this method is taken due to part of source videos were shot by a drone which is far away from the cyclist. These new cropped images provide a closer view of the target cyclist. Figure 3.8 and Figure 3.9 shows one example.

![Figure 3.8: Before cropping.](image)

![Figure 3.9: After cropping.](image)
Brightness and contrast adjustment

After this process, the overall image gets brighter and relatively more details in the darkness is revealed. In the meanwhile, increased contrast sharpen the image, i.e., the difference between colors gets more obvious. Figure 3.10 and Figure 3.11 shows one example. This method is suitable for both scenarios:

- When the drone follows the cyclist and cast light, the cyclist is more distinguishable.
- When the drone loses track of the cyclist, it turns out to be easier to recognize based on the bike light.

![Figure 3.10: Before adjustment.](image)

![Figure 3.11: After adjustment.](image)

3.1.6 Dataset Description

The final augmented dataset consists of 1681 images which are various in resolution, lighting conditions and shooting angle. To avoid data type imbalance or the accumulation of images in sequential order in one single dataset, these 1681 images are shuffled in a proportion of 8:1:1. Therefore in the end there are 1345 images for training and 169 images for validation. Table 3.1 shows the training data distribution.

<table>
<thead>
<tr>
<th>Data Classification</th>
<th>Description</th>
<th>Total</th>
<th>Description</th>
<th>Total</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Image Size</strong></td>
<td>Cropped(350 × 350)</td>
<td>292</td>
<td>Original</td>
<td>1389</td>
<td>320 or 350</td>
</tr>
<tr>
<td><strong>Lighting Conditions</strong></td>
<td>Dark</td>
<td>670</td>
<td>Light</td>
<td>1011</td>
<td></td>
</tr>
<tr>
<td><strong>B &amp; C Adjustment</strong></td>
<td>Adjusted</td>
<td>194</td>
<td>Original</td>
<td>1487</td>
<td>B 35%, C 1.5</td>
</tr>
</tbody>
</table>

Table 3.1: Training dataset classified by different features.
Given the goal of this thesis work is to apply the model in the dynamic scenarios, i.e.,
the environmental conditions varies a lot such as lighting and irrelevant object, it tends
to be a more effective approach to assess the model’s robustness with videos instead of
images. Therefore, during the testing, three different video clips are used for evaluation.

3.2 Simulation Tool

For this thesis project, the test was planned be conducted on a simulator named AirSim,
which is an open source plugin built on a real-time 3D rendering platform Unreal
Engine with excellent visual effects, Figure 3.12 demostrate the interface of a multirotor
in the template background, Figure 3.13 shows the switched view from one of the
cameras. In addition, various APIs are offered to enable users to adjust the weather,
communicate with vehicles, retrieve data etc. Therefore, it is worth considering for
experiment computer vision task on vehicles such as drones or cars.

![Figure 3.12: Interface of AirSim running a multirotor in the Landscape Mountains
template background.](image1)

![Figure 3.13: The view from the multirotor after switched camera.](image2)
However, due to various limitations, testing was not conducted on this platform. Check the description in section 5.2 for details.

### 3.3 Evaluation Metrics

#### 3.3.1 Precision and Recall

Before explaining these indicators in the form of formulas, these important parameters of confusion matrix for binary classification are going to be introduced first.

- **True Positive (TP)**: correct prediction. Detected the labeled object.
- **False Positive (FP)**: wrong prediction. Detected, but it is not the labeled object.
- **True Negative (TN)**: correct prediction. Doesn’t detected, the labeled object does not show up neither.
- **False Negative (FN)**: wrong prediction. Doesn’t detected, but the labeled object exists.

Precision shows how accurate the model is when doing the prediction, it is the ratio of right-predicted bounding box to all bounding boxes. It is shown in Equation 3.1:

\[
\text{precision} = \frac{TP}{TP + FP} = \frac{TP}{\text{All Positive Detection}}
\]  

(3.1)

Recall indicates how well the model performed in reality, which means here the number of correctly predicted boxes is compared to the total amount of the target objects. It is shown in Equation 3.2:

\[
\text{Recall} = \frac{TP}{TP + FN} = \frac{TP}{\text{All Ground Truths}}
\]  

(3.2)

These equations also suggest that precision and recall constrain each other since after training the same dataset, the increase in the number of positive samples also means the decrease in negative samples.

#### 3.3.2 Intersection over Union

When the model thinks one region might contains the object, a bounding box is proposed. Assume there is a predicted box and a ground truth box, the Intersection over Union (IoU) value is the proportion of the overlapping area to the combined area of both boxes, ranging from 0 to 1. An IoU value of 0.5 is commonly considered as a moderate threshold in object detection tasks. A result below this threshold suggests a poor overlap and detection, while a result with higher value like 0.7 can indicate an accurate localization. Therefore, the higher the IoU value is, the better the precision is. IoU threshold is also commonly used as a part of the evaluation to tell whether the prediction is correct or
wrong [30]. This concepts is shown in Equation 3.3 and demonstrated in Figure 3.14, Box A represents a ground truth box and Box B represents one of the bounding boxes.

\[
\text{IoU (BoxA, BoxB)} = \frac{\text{BoxA} \cap \text{BoxB}}{\text{BoxA} \cup \text{BoxB}}
\]  

(3.3)

Figure 3.14: Simple explanation of IoU.

3.3.3 Non-Maximum Suppression Algorithm

During the detection, boxes with different score numbers might be generated around one same object. Confidence score of each proposed bounding box is computed based on mean average precision at IoU threshold. Then non-maximum suppression algorithm is applied to filter out redundant overlapping bounding boxes and eventually extracts the most accurate result.

3.3.4 AP and mAP

Average Precision (AP) is the area computed according to P-R curve on the interval [0,1]. If there is multiple classes, mAP stands for the AP of each class divided by the total number of classes. It is an important indicator to evaluate the ability of the object detection model. The Equation 3.4 and 3.5 below describe respectively.

\[
\text{AP} = \int_0^1 P(R) \, dR
\]  

(3.4)

\[
\text{mAP} = \frac{1}{N} \sum_{i=1}^{i=N} \text{AP}_i
\]  

(3.5)
3.3.5 Parameters Setting

The parameters in the experiment is presented in Table 3.2.

<table>
<thead>
<tr>
<th>No.</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Epochs</td>
<td>300</td>
</tr>
<tr>
<td>2</td>
<td>Batch Size</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>Image Size</td>
<td>$640 \times 640$</td>
</tr>
<tr>
<td>4</td>
<td>Learning Rate</td>
<td>0.01</td>
</tr>
<tr>
<td>5</td>
<td>Optimizer</td>
<td>SGD</td>
</tr>
</tbody>
</table>

Table 3.2: Parameters for model training.

3.3.6 Experimental Configuration

The environment of the experiment is presented in Table 3.3.

<table>
<thead>
<tr>
<th>Item</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>12 vCPU Intel(R) Xeon(R) Silver 4214R CPU @ 2.40GHz</td>
</tr>
<tr>
<td>GPU</td>
<td>RTX 3080 Ti(12GB)</td>
</tr>
<tr>
<td>Framework</td>
<td>Pytorch 1.11, CUDA 11.3</td>
</tr>
<tr>
<td>Language</td>
<td>Python 3.9.3</td>
</tr>
</tbody>
</table>

Table 3.3: Experimental Configuration.
4 Results & Discussion

4.1 Training Results

As section 2.3.5 has mentioned, loss function is a crucial role throughout training process. The result graphs visualized the difference between the predicted result and the true value over the training and validating iterations.

A YOLO output tensor consists of three components:

- Bounding box coordinates(x, y, w, h).
- Objectness score. They indicate the confidence that the bounding box contains an object.
- Classification scores. They represent the probability distribution and processed through softmax activation function to ensure that they sum to 1. In this project, this part will not be discussed since there’s only one class.

4.1.1 The best result

After several data processing attempts, the augmented dataset works well. The fluctuations in localization loss values and objectness loss values during training epochs is depicted in Figure 4.1. During the training process of the first 100 epochs, there are several sudden increase in individual loss vaules, but they quickly return to the steadily declining curve and complete convergence. This may be caused by the model suddenly learning an object that is very different from the previous batches. After about 250 epochs, the curve tends to ease and there is no obvious change, which means that the parameters of the model have reached the optimal level and the learning process is basically finished.

![Figure 4.1: Training loss-best result.](image1)

![Figure 4.2: Validation loss-best result.](image2)
This trend is favorable since the goal of calculating loss values is to optimize model parameters. As the training time increases and the model gradually fits the data set, the gap between the predicted values and the true values is expected to decrease.

Figure 4.2 is the case for validation dataset, which has the similar trend. Different from the training loss which guides the model learning, the validation loss evaluates generalization ability towards unseen data. In addition, during this process, parameters will not be updated. In this figure, the validation loss also presented an expected decrease, then goes steady. It means the model has successfully captured feature in training dataset and has the ability to identify targets in validation dataset. Meanwhile, this indicates the robustness of the model.

The result graph Figure 4.3 is about the valuable comprehensive indicator mAP. The indicator mAP@0.5 represents the average mAP with a threshold greater than 0.5 while mAP@0.5:0.95 represents the average mAP at different IoU threshold (from 0.5 to 0.95 with a step size of 0.05). After several data processing attempts, the augmented dataset works well. The overall trend shows a steady increase and then leveling off. In the end the model achieves 98.88% mAP@0.5 and 67.89% mAP@0.5:0.95. This suggests the model can generally detect object, but it is not always the case that the predicted bounding boxes overlap highly with the ground truth boxes. This is conceivable based on the fact that our data set is not very large and it contains objects which vary in features.

4.1.2 Analysis of Previous Experiments

The unshuffled dataset

When creating this new batch of datasets, I believed it was efficient to select valid data then assign them for training/ validation/ testing use directly. This also should be some kind of randomness, although the assignments were not completely random. At least the distribution would not be quite uneven.
Results & Discussion

After training for 300 epochs, the result appeared unexpectedly poor. As can be seen from this plot in Figure 4.4, the trend is rising but almost every indicator fluctuates greatly and it’s not getting stable. Based on the most recent training results, which only went through approximately 400 images, the size and complexity of this new dataset may had a certain impact on the results. In the end, the mAP@0.5 is 66.83% and the mAP@0.5:0.95 is only 26.8%.

Another slightly disturbing thing is the non-smooth validation loss curve in Figure 4.5, though the overall trend suggests the model is adapting towards an ideal direction. It is most likely due to data imbalance, which might result in the current model not having learned enough about objects in certain types of scenes.

After this training, I reorganized the dataset with a real shuffling tool: Python library. It turned out to be effective.

![Figure 4.4: mAP-unshuffled dataset.](image)

![Figure 4.5: Val loss-unshuffled dataset.](image)

**The oversized batch**

If the former result is considered poor, this one is a little bit disaster. I was trying to increase the batch size for more efficient training, so the batch size was once increased to 32. The explanation of this unstable plot could be related closely to this.

Batch size refers to the amount of data that the model will go through, after the forward propagation of this batch, the model starts back propagation and updates weights. If there’s 1280 items in all, then one epochs means 40 iterations like this will take place. When each update involves more data, the model is more sensitive to the content.

Consequently, another assumption might be that the number of frames selected from each video is slightly different due to the specific feature of videos. Plus the dataset size is probably relatively small. The final results of mAP (plots in Figure 4.6) is better than the last case, mAP@0.5 rises to 77.62% and mAP@0.5:0.95 increases almost one half, 43.71%. It is likely that the shuffled dataset balanced the datatype in different sets.
The loss curves in Figure 4.7 might happen when the size of the dataset, complexity and batch size are not compatible, especially in some cases the number of pictures is very limited. The oversized batch will result in more fluctuating updates, or even worse, overfitting. The curves jump to a relatively higher value several times during the first 100 epochs, and until the end they have not achieved good convergence.

4.2 Testing Results

4.2.1 Result from the test dataset

We firstly evaluated the performance with the test dataset of 167 images. It recognized the cyclist in each image, as Figure 4.8, 4.9, 4.10 and 4.11 presented.
However, there are few misjudged images, all of them can be categorized to "False Positive", which means the model detected something but it is not the target. When we tried to analyse Figure 4.12, the reason is apparently related to the area that we put bounding boxes when we annotated data. The other example Figure 4.13 is a little bit interesting but it is understandable due to the pattern is similar to the shape of most cyclists in the dataset, especially under worse lighting conditions. In addition, among several consecutive frames, only this one was misjudged.
4.2.2 Result from the unseen video

These videos for testing the generalization. The trained model performed as expected, successfully detecting the cyclist until the front half was entirely blocked by the bridge. After cycling past the under bridge section, tracking can be resumed. Figure 4.14 and Figure 4.15 are example screenshots.
Screenshots Figure 4.16 and Figure 4.17 are from another testing video, the drone is relatively close to the cyclist. The size of prediction box changes with the varying light conditions, consistent with the way of data annotation for model training use.
4.3 Discussion

While preparing the dataset and labeling it, we realized that in most cases the model was identifying objects with such features rather than a single “person”: A strip-shaped target, with the beam of light projected by a bicycle’s headlight at the front, a cyclist who sometimes wears a reflective vest in the middle, and then the taillight. The labeled area varies depending on visibility, as shown in Figure 3.7. Therefore, it’s not surprising when error detections like the one in Figure 4.13 appear.

To answer RQ1:

Yes, using real-world videos captured by our drones to develop a model based on YOLO for tracking purpose is a promising approach. There are more discussion about the source videos. Since the drone responsible for filming is in most cases above the drone responsible for tracking and throwing light, the overall feature of the source video set is that the cyclist is relatively small in the frame. Despite the use of data enhancement technique, the clarity of goals cannot be more significantly improved. Extracted single frame is already quite difficult for unrelated people to identify. Be owing to the fact that source videos closer to the cyclist are more limited, they are reserved for testing and training is based on the rest. Existing literature suggested that a feasible further enhancement might be adding a new set of anchor boxes for detecting smaller objects. Simultaneously, it is worth attempting to train the model using more videos captured from a shorter distance.
To answer RQ2:

It works well in the specific environment that this project aims at, which is shown in section 4.2 Testing Results. Analysing the test results, the original model performed favorable since the mAP@0.5 is quite close to 100%. Moreover, from the visual perspective, the model can indeed achieve consistent and stable identification of cyclists, despite interference from obstructions along the way. This success can be attributed to effective data cleaning and labeling.

Overall, the experiment results from this study support the answer to the research questions posed at the beginning. The pre-trained weight of YOLOv7-tiny, the lightest model in the same version, can be used for developing a model based on our customized dataset, and it performs properly in testing.
5 Conclusion & Future Work

5.1 Conclusion

Motivated by the widespread integration of drones in the most recent years, the project Skara Skyddsängel is brought up for drone-assisted cyclist tracking along a 20km unlit bike lane. The initial experiments were conducted to explore the feasibility of GNSS with a mobile application. However, the drone loses track of the cyclist occasionally due to the inconsistent signal strength. Consequently, this thesis project is proposed for the implementation of an object recognition model to enable robust drone tracking in darkness.

After reviewing relevant literature, we chose to experiment with the light-weighted YOLO-v7 tiny model. Due to the unique application scenario, we deployed drones to capture videos in the spective region instead of using existed dataset from open-source platform. Thereafter, a selected collection of videos was used for frame extraction to create our dataset.

Following multiple rounds of data screening and data augmentation implementation, an appropriate dataset was prepared for final training. After numerous comparative training experiments, we successfully obtained a model with high accuracy when evaluated by test dataset and previously unseen videos. The statistical analysis and the test results indicate that the model is capable handling cyclist detection and identification.

5.2 Potential Future Improvement

Future work can take the following points into consideration:

- The first challenge arises when it comes to the simulation method. The current model is trained based on real-life videos collected by drones. Therefore, the virtual environment also supposed to be as close to the reality as possible to ensure efficiency. As already mentioned in Section 3.2, simulation studies are considered as a major future work.

- The current training results are more inclined to the situation of single-cyclist tracking with out the assistant of GNSS app communication. This may face limitations when confronted with situations involving multiple cyclists with variations in
speed and direction. One potential solution might be leverage the coordinates of detected bounding boxes to deduct the appropriate optical motion to make sure the drone is still following the same target cyclist.

- The latest version up to now is YOLO-V8, which was released in January 2023. We assumed YOLO-V7 is more well-established, so we decided to take a further look.

- While we have tested and validated the algorithm with the available dataset, implementation and integration into drones remain to be investigated. This forms another interesting future work.
Literature


