Department of Mathematics
and Computer Science

University of Southern Denmark

Master Thesis in Environmental Data Science

Habitat Surveillance augmented through Unoccupied Aerial Vehicle Computer Vision

Author
Jakob Jaensch Rasmussen
Exam nr: 92412296
Jakra16@student.sdu.dk

Supervisors
Peter Schneider-Kamp
Niels Svane
Naeem Ayoub

May 31, 2022

Abstract

Overvågning af dansk habitatnatur er en tidskrævende opgave, der benytter sig af manuel arbejde udført af udpegede eksperter, der i dokumentationsfelter af 20m^2 identificerer forekomsten af plantearter og andre habitatkendetegn. Indsamlede data benyttes til at dokumentere og digitalisere habitattilstanden ved, at ekstrapolere dataene til at gælde for det komplette habitatområde. Droner er i stand til at tage højopløsningsbilleder, og kan dække mange hektar under deres flyvetid for ét enkelt batteri. Dette studie fremfører en serie af objektdetektorer baseret på YOLOv4 arkitekturen, der er i stand til at detektere de to invasive arter kæmpe bjørneklo (*Heracleum mantegazzianum*) og rynket rose (*Rosa rugosa*), samt estimere populationernes dækningsgrad med høj nøjagtighed. Detektorer er også udarbejdet for strandkål (*Crambe maritima*), som bliver påvist detekterbar af algoritmen, men ikke med en tilstrækkelig nøjagtighedsgrad (52,8%) der kan understøtte overvågningen af hele populationer. Én detektor for henholdsvis kæmpe bjørneklo og rynket rose fremføres som optimerede modeller. Detektorerne viser begge høj præcision for estimering af dækningsgrader, henholdsvis 87,9% for kæmpe bjørneklo og 83% for rynket rose. Bjørneklo kan lokaliseres med en nøjagtighed der resulterer i en F1 score på 0,9, hvorimod rynket roses F1 score er 0,64 og lokalisering af enkeltstående individer er derfor tvivlsom. Evnen til at detektere populationen som helhed forbliver dog tilfredsstillende. Der forelægges en arbejdsstruktur for, hvordan detektorerne kan benyttes til at detektere objekter, simultant med der flyves med dronen i felten, samt en automatiseret digitalisering af detektioner i et geografisk informationssystem uden for felten. Detektorerne skal afhjælpe den danske indrapportering af statussen for habitatnatur under Den Europæiske Unions Habitat Direktiv, og vil som koncept kunne implementeres i samtlige indsatser i de øvrige EU-nationer, der indgår i samarbejdet.

Contents

Introduction

The European Union's Habitat Directive declares that member states are to capture the temporal status and trends of their country's habitat types and report on these to the union every six years [EU, 1992]. Traditional methods for monitoring habitats rely heavily on expert *in situ* inspections of high time consumption with low spatial accuracy in assessments as observations often are extrapolated to represent vast areas. Supplementing *in situ* findings through *in silico* methods have become increasingly popular with the rapid technological growth within remote sensing. The use of mapping software, satellite imagery, and image segmentation is now widespread within the environmental sector [Blaschke, 2010]. The increased spatial coverage and temporal traits of captured imagery allow for large-scale inspections that with traditional *in situ* methods would prove time-consuming and unfeasible.

A hurdle in current management is the lack of spatial resolution in imagery as the classification of habitat types can demand that certain plant species, rock-, and sediment types are present [DEP, 2016a,c]. Utilizing a combination of *in situ* and *in silico* methods to mitigate each other's weaknesses through the aspect of ground-truthing (verifying and tweaking *in silico* results through *in situ* observations) is therefore common in present surveillance protocols [DEP, 2016b]. The inherent importance of *in situ* inspections can be seen as the limiting factor of the rate at which monitoring- and annotation of habitats occur. Improving on this aspect together with a revision on how management is performed *in silico* can lead to member states increasing their efficiency.

Unoccupied aerial vehicles (UAVs) have gained great traction in their implementation in several sectors as drivers of innovation within remote sensing and robotic automation [Mohamed et al., 2020]. UAVs with various sensors have already been tested and implemented within ecological- and environmental solutions thanks to their vast potential [Berni et al., 2009, Bryson et al., 2013, Mohamed et al., 2020, Svane et al., 2021]. Remote sensors have gained traction through machine learning, where computer vision seeks to promote autonomous operations and intelligent innovation in augmenting both everyday and complex professional tasks, promoting computers to ascend to and beyond human-level visual perception [Al-Kaff et al., 2018]. Image- and video material allows for machines in real-time or post-capture processing to derive information through machine learning algorithms revealing relations, information and trends which are often incomprehensible by human intuition alone [Voulodimos et al., 2018].

Incorporating UAVs into computer vision allows humans to ascend into altitudes where visual inspection can be augmented in real-time, increasing spatial reach and motivating users to increase the frequency of conducting operations. Unlike airplane or satellite imagery collection, UAV flight can be executed manually or autonomously a few meters off the ground in an easy and cost-efficient manner, leading to detailed high-resolution imagery. Is the use of UAVs as a method of inspection the high-resolution data foundation needed to increase accuracy and efficiency in ecological habitat surveillance? The Danish Environmental Protection Agency (DEPA) is interested in how they can benefit from high-resolution imagery in habitat surveillance operations. This study will be carried out in collaboration with DEPA and constitute a proof-of-concept in DEPAs digital innovation strategy. The DEPA is interested in applying several sensors for use in UAV surveillance. This study will emphasize the use of RGB cameras as these are standard-issued equipment for most UAV platforms.

The aim of this study is to (1) collect data using UAVs in Danish habitat nature and (2) construct computer vision algorithms that will aid in habitat surveillance. (3) Algorithm performance will be comparatively evaluated with respect to current DEPA surveillance efforts, and (4) assessed in an evaluation on performance and prospects.

Background

The State of Nature

Globally nature has been under immense pressure. As the human population grows, so does the need for increased infrastructure, agriculture and other resource production sites [EEA, 2020]. This growth demands the elimination of natural habitats and pressures remaining nature through pollutants such as chemicals, lights, sounds, and other foreign invasive factors[EEA, 2020, p. 72]. The European Habitat Directive aims to prevent this trend and holds member states accountable for the state of European nature [EU, 1992]. Through the directive, member states report their findings, and the European Environment Agency compiles the results into a status report on the state of European habitats [EEA, 2020].

In the latest report, 2013-2018, from the European Habitat Directive, the results reported for Denmark has highlighted that only 5% of land and water habitats were in a good preservation status, with the remaining 18% in an adverse state and 77% in a strongly adverse state [DCE, 2019]. Compared to the previous status report, the trend in habitat states shows that only 5% was in positive growth and 13% remained status quo [DCE, 2019]. The remaining habitats revealed that 33% were in decline, and for 50% the trend was unknown, probably due to it not being recorded for the previous report period [DCE, 2019]. The trend is clear; Danish nature is in decline and like so in the remaining member states of the European Union, with only 15% of habitats being in a good conservation status [EEA, 2020]. The lack of proper habitat care is not a national problem but an international one and globally a significant problem in preserving ecosystems. Upholding the goals outlined in the United Nations sustainable development goals on climate action, life on land and -below water is thus globally at risk [UN, 2022].

Current Habitat Surveillance

Field protocol inspections

Danish habitat types in open sunlit areas are classified through inspection with a DEPA-appointed expert *in situ* equipped with the Nature key protocol [DEP, 2016a]. The protocol is a Danish tool for examining national habitat types and the findings are referred to with the European Union Habitats Interpretation Manual which is the legally binding classifier for member nations in the European Union (EU) [EU, 2013]. Findings are verified prior to digitizing areas as polygons in a geographical information system (GIS) [DEP, 2016b] for the report using the Danish habitat description curriculum [DEP, 2016c]. The Danish habitat description curriculum has been constructed in response to the EU habitat interpretation manual as:
"The manual is easier to misinterpret than to understand" [DEP, 2016c, p. 1]

This statement should be seen as a result of the judicial level at which the EU handout is written, with its character being legislative rather than that of a field protocol. DEPA describes difficulties in performing proper annotations as *in situ* inspections are representative of small survey areas, not the complete habitat, as is often the case when surveying large habitats [DEP, 2016a]. The survey areas inspected by DEPA are called "documentation sites". These sites are selected depending on if a change is observed in satellite imagery in relation to previous habitat inspections. If no noticeable change is recorded, then the previous sites are reused for the survey. Documentation sites are circles of five meters in diameter where the vegetative- and species distribution is examined. The habitat type is denoted from the findings within the documentation field in accordance with the related field protocols [DEP, 2016a,c]. The annotation of habitat types is highly dependent on the species composition or presence of single species within the documentation sites.

Digitization efforts

As described habitat classification is digitized and stored in a GIS system as shapefiles, geometric visual representations of entries in a tabular data frame. Habitats are digitized as polygons with their span based on the documentation site findings in relation to the surroundings visible on satellite imagery. Any endangered or invasive species identified outside the documentation site is supplementarily noted for the annotation to keep track of the presence and distribution of these sensitive species [DEP, 2016b]. DEPA utilizes GIS databases for the management of most of their data on Danish nature and the annotations for habitat nature are publicly available in open GIS interfaces [Miljøportal, 2022]. This leaves habitat annotation stored as layers of polygons covering the classified areas. The extent of the polygons is determined by inspecting aerial imagery in conjunction with previous years' annotations in light of the new findings from documentation sites [DEP, 2016b]. The documentation site itself is additionally registered for future use and to keep historical records of inspections available.

DEPA notes that for small scale variations observed *in situ* in so called mosaic landscapes, for example beach meadow, they strive to include as many habitat types as possible under *one* polygon with habitat coverage denoted as percentage-wise distribution [DEP, 2016a]. Tackling mosaic nature in this way leads to a descriptive detailed annotation but visually and spatially poor annotation. It is evident that the task of accurately classifying habitat types is daunting. It is difficult to fulfill the expectations the EU has put forth as it currently requires extensive *in situ* surveys with current aerial imagery being insufficient in standing alone as documentation.

Another shortcoming of current efforts is the lack of evidence and access to historical data as only protocols and perhaps later digitized annotations from human-vision inspections remain as evidence of observations [DEP, 2016b]. These observations can then be supported by old aerial imagery, but the resolution is too low to identify anything smaller than a tree or a car within the imagery. It goes without saying that with today's capabilities within analytics, observations by human vision alone do not make up a solid data foundation as the findings are unable to be backed up by any material evidence. A shortcoming acceptable in the past but from 2010 and onward this lack of quality documentation becomes difficult to ignore. The lack of spatial detail and historical backlogging in decision-making is a pitfall in the habitat surveillance program stemming from the time costs and manpower needed in inspecting all habitats. Ideally, high-resolution imagery should be available for surveyed habitats and in a perfect world using these images alone it should be easy to identify habitat identifiers allowing for easy classification. This would still require immense work manually inspecting all imagery, but analytics boosted through machine learning methods could feasibly close the gap in cutting out manual inspections securing instead a more automated data-oriented workflow. This is not unfeasible and investigating new ways of approaching nature surveillance can yield opportunities to augment or even surpass current efforts in surveillance [Svane et al., 2020].

Consumer UAV platforms

Unoccupied aerial vehicles come in several different shapes and sizes, and the term therefore covers everything from weather balloons to large militarized surveillance air crafts, [Al-Kaff et al., 2018]. Today, most UAVs are known under the collective household name "drones" and relate to small UAVs controlled using remotes, often coupled with a display conveying visual information of the UAV's surroundings. RGB cameras are remote sensors capturing the red, green and blue color spectrum manifesting as depictions of the world as experienced by human vision- in other words, your everyday camera. RGB cameras are considered a standard on most UAV setups. Most UAVs are rotorcrafts achieving flight by the means of one to several rotor blades. Most rotorcrafts benefit from being able to stand still mid-flight at variable altitudes allowing sensors to operate under controlled stable conditions. Some UAVs are additionally equipped with camera gimbals that

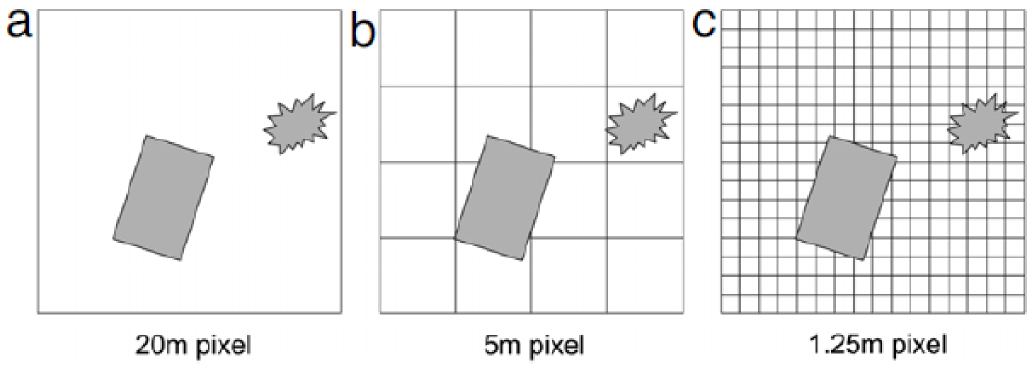

Figure 1: *The relationship between object representation and spatial resolution [Blaschke, 2010, p. 3]. (a) Low spatial resolution with pixels being significantly larger than objects. (b) Medium-scale spatial resolution with pixel and object sizes being of the same order. (c) High spatial resolution sees pixels being smaller than represented objects and allows for detailed representation of objects.*

stabilize the camera to a pre-calibrated stable baseline allowing for UAVs to fly at great speeds and simultaneously record imagery steady and seamlessly.

Several manufacturers supply consumer-grade UAVs, and one such is DJI which is responsible for the popular Phantom drone series [DJI, 2022]. The Phantom series has seen implementation in commercial applications, research projects and general recreational use. The UAV series is probably the platform most people think of when hearing the word "drone". DJI's Phantom drones are quadrocopters equipped with a high-resolution RGB camera mounted on a motorized gimbal. The platform can perform automated flights using a flight planner companion app and allows for easy flight execution by the user. The standard battery is rated for about 30 minutes of flight on one full charge, and being compact allows the user to bring several batteries on flight missions. The versatile phantom series has been documented in literature as a reliable platform for use in event photography, archaeological site inspections [Holness et al., 2016], marine vegetation mapping [Svane et al., 2021], photogrammetry [Carbonneau and Dietrich, 2017], forest surveillance [Mlambo et al., 2017] and other fields of use.

Purposing aerial imagery for image analysis

Capturing aerial imagery from a UAV is mainly performed in one of two ways; top-down at an angle of -90° horizontally, an angle known as nadir view, or free-flowing in variable angles that most would describe as panoramic views. UAV imagery at nadir view is equivalent to that of airplane photography or satellite imagery but can instead achieve greater spatial resolution in direct comparison. Spatial resolution is promoted by having the altitude of operations possible at heights of 5-100m. The cost of low altitude surveys is a lack of spatial coverage relative to operation time for other aircrafts and satellites. Spatial resolution relates to the real-world linear dimensions of pixel units within the imagery. When capturing images of objects of interest, spatial resolution is important as it determines how clearly one can see and identify the objects in question. Low spatial resolutions covering large linear dimensions lead to coarse or even insufficiently detailed imagery, Figure 1a-b, while high spatial resolutions see pixels covering short linear dimensions leading to detailed imagery, Figure 1c.

To capture quality imagery purposed for analyses like those in deep learning, the flight operator needs to consider several factors. Calm weather is preferred as the low wind allows the drone to stabilize and is also required for safe operations. Capturing imagery of buildings, roads and other infrastructure is relatively straightforward as these inanimate constructions remain unaltered no matter the weather or climate conditions. Plants being stationary non-inanimate objects propose difficulties as seasonality determines in what state these objects will be recorded. Most plants have flexible parts, be it branches, flowers or the plant in its entirety, and mild winds are preferred when capturing imagery to keep their poses uniform. Plants are closely positioned together, and the overshadowing of objects of interest is very likely. In the light of deep learning, shadows propose a challenge as algorithms can tend to fit non-unique shadows for certain classes if these are present within the training data. Manual inspection of imagery is also hampered by the presence of shadows at times, as some objects may become intelligible to locate within the images. Capturing imagery at midday when the sun is at its highest is preferred to reduce the overshadowing effect as all objects are in full light and shadows minimized. Depending on the object of interest, the spatial resolution should be optimized by calculating the proper altitude for capturing material of a sufficient spatial resolution.

Computer Vision

Defining vision

Computer vision is an integral part of many UAV applications. The common objective of flight is to remotely observe using an RGB camera or other remote sensors in perspectives inaccessible to humans. Visualizing the sensor's input on a display is not considered computer vision as it requires the system to interpret what is pictured and derive meaningful information. Data gathering is therefore not to be confused with computer vision as visual information is gathered and interpreted by the operator. Automated flight using flight planner application software is neither considered a true automated flight *based* on computer vision as it instead is automated using preplanned routes, and the presence of the camera sensor is not a prerequisite.

Proper computer vision applications can be applied to commercial UAVs, and several different applications have already been successfully proposed and implemented [Al-Kaff et al., 2018]. Among these applications are autonomous takeoff and landing [Jung et al., 2015], route planning [Yang et al., 2014], tracking [Zhao et al., 2013] and many others. Object detection is one of the underlying keys in promoting autonomous operations. The ability to detect and classify objects allows the system to interpret its surroundings and make decisions with vision as the main driving force.

Convolutional neural networks

Detection of objects and classifying images is a vast field within computer science under the collective banner of computer vision. In 1980 the world was introduced to the precursor of convolutional neural networks (CNNs) via the Neocognitron, a type of neural network designed with inspiration from visual nervous stimulation in vertebrates [Fukushima, 1980]. It introduced the core components of modern CNNs; convolutional layers and downsampling layers. For grid arranged data, like an image's pixel values represented in a matrix, convolutional layers cover larger grid sections via a kernel window, computing several values at once while outputting only one value. The kernel window strides across the grid and comparatively reduces the output grid size by taking several input grid values into account when computing. This trick essentially speeds up computation and reduces the dimensionality of processed data, further promoting speed. Downsampling layers, commonly referred to as pooling layers, pool together grid values, reducing input dimensionality by (*usually*) averaging values within a set window size, promoting signal intensity while reducing data complexity.

In 1998 CNNs were popularized when LeNet-5 was constructed, an efficient handwritten digit classifier operating efficiently on complex data utilizing a relatively small network size [Lecun et al., 1998]. CNNs were now a force to be reckoned with within the field of computer vision, and in 2012 the ImageNet challenge was won and has since been won by CNNs [Krizhevsky et al., 2012]. CNNs are present in almost all state-of-the-art image analysis architectures, and in general, they all infer at greater speeds and greater accuracy compared to ordinary neural networks.

Image analysis can be categorized into three major categories: image classification, image segmentation and object detection. Image classification classifies the entire image with a label. Having one gross label for an image is not always beneficial, as multiple objects can be positioned within the image grid. Introducing image localization allows for multi-class labeling for imagery while also increasing the complexity of the network. Image segmentation segments the image allowing for multi-classification of constructed regions within imagery in areas of interest. Object detectors are similar to segmentation algorithms in localizing objects of interest but do not draw a region with a determinable area around objects but rather frame them using a bounding box (bbox). One should consider what model type is the most appropriate for the task and imagery at hand. Multiple objects of interest will be presented within UAV imagery as it spans large areas. An implementation emphasizing localization would be the appropriate choice for UAV imagery to locate objects of interest accurately.

The YOLO Algorithm

There are two important factors in object detection; 1. accuracy- and 2. computational speed of classification. Several CNN-based algorithms have been proposed. Several of them perform sufficiently, but no single algorithm architecture has proven prevailing as the preferred algorithm. The "You Only Look Once" algorithm, better known by its acronym YOLO, is one of these popular computer vision algorithms. YOLO is an object detector and has gained traction since its inception in 2016 [Redmon et al., 2016] as it swiftly infers several detections within imagery at speeds that allow for inference on video material. Identified classes are intuitively indicated in the frames by drawing a bbox around the object and displaying the class label accompanied by the confidence level in percent. YOLO's great inference speed, outperforming image segmentation algorithms like RCNN [Bochkovskiy et al., 2020, p. 10], in combination with low technical requirements, makes it a suitable detector for image localization on video material and large and noisy imagery.

YOLO is intended for use as an object detector, and usage of the algorithm in stationary video sources (CCTV, etc.) is favorable as the field of view and variation within the visual area is little to none. Since 2016 YOLO has seen wide application in various tasks as a counter for pedestrians [Menon et al., 2021], -face masks during the covid pandemic [Loey et al., 2021], a fruit ripeness determinant [Tian et al., 2019], a tissue anomaly detector in X-rays [Al-masni et al., 2018], a traffic monitor [Chen et al., 2020], an aerial component inspection aid for infrastructure [Ayoub and Schneider-Kamp, 2021] and several other relevant object detection related applications.

The algorithm is relatively easy to use and set up but challenging to adapt and master. Research into the optimization of YOLO and new iterations on the architecture have been proposed since 2016, along with a compressed, faster inferring, and less demanding model type being introduced, the so-called tiny models [Adarsh et al., 2020]. The base algorithm has since 2016 been optimized, and today is widely considered to be in its fourth version state [Bochkovskiy et al., 2020]. Some stakeholders argue that YOLO version five has arrived and is seeing greater improvements; this claim, however, stands without any peer-reviewed documentation[Jocher et al., 2021, *No paper or peer review*].

In purposing YOLO for a specific detection task, one must train the algorithm on new data. These data need to be labeled, and this can be a daunting task if no public data is available, as gathering and labeling images of classes takes immense time. There are six phases in purposing YOLO for classification and detection of custom classes:

1. Select a YOLO architecture with appropriate input size.
2. Label data for custom classes.
3. Augment data set if the set size is not satisfactory.
4. Split the data into a training-, test- and validation set.
5. Train the algorithm on the training set and evaluate the accuracy using the test set.
   - Augment data if performance is not satisfactory.

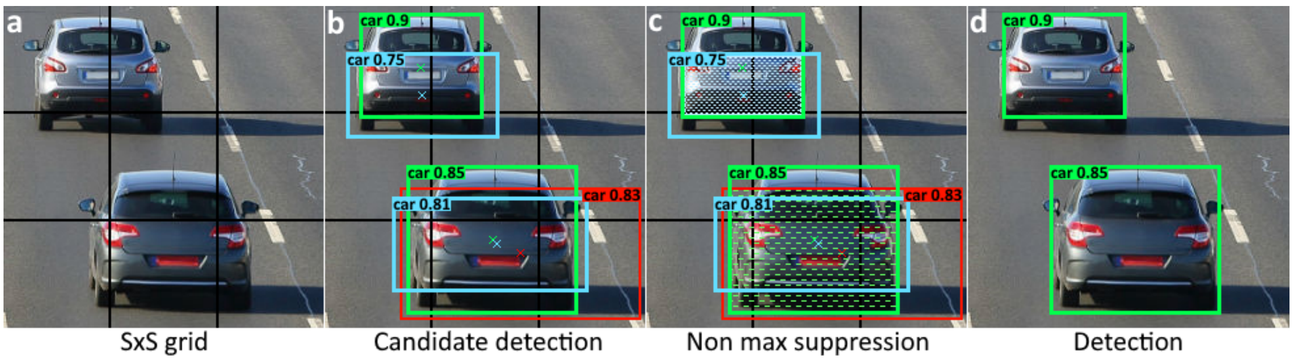

Figure 2: *The process of detecting objects in the YOLOv4 algorithm. (a) The input is segmented into SxS regions. (b) For each region, detection candidates are marked within regions. (c) Based on regions, the intersection of union between bounding boxes is calculated. (d) Using non-max suppression, the best detection candidate is kept and visualized in imagery as detected objects.*

6. Validate the final classifier on the validation set.

If a sound data foundation is secured, training the YOLO algorithm should be relatively straightforward with powerful hardware at the researcher's disposal.

To understand how YOLO works as an object detector, we will look at a quick walkthrough of the detection process. When detecting YOLO segments input imagery into an $S \times S$ grid, Figure 2a. For each grid cell, if an object is suspected to be present, YOLO draws a bbox and registers properties of centroid coordinates and bbox width and height. The centroid of the objects decides what region grid cell is responsible for detecting and classifying the object even though the bbox may span multiple regions, Figure 2b. YOLO predicts whether a class is present or not and labels the bbox with a confidence value from 0.0 - 1.0, not current to undoubtedly present.

Multiple bboxes can be drawn over the same object, and the same class of objects can be present within the imagery at several locations numerous times. YOLO determines and locates individual objects by considering the intersection over union (IoU) between bboxes at a set threshold, often 50%, using non-max suppression (NMS), Figure 2c. For several bboxes, the bbox with the highest confidence value is selected, and all neighboring bboxes with an IoU above the threshold are dropped as these share a high union but at a lower confidence level – effectively, this suppresses non-maximum confidence detections, Figure 2d. Objects of the same class having centroids within the same region can essentially be distinguished as their IoUs should fall below the threshold.

YOLOv4

YOLOv4 was proposed in 2020, building on previous breakthroughs with YOLOv3. It successfully aimed to improve inference speed and classification accuracy while lowering the requirements of training such an algorithm to a level where consumer-grade graphics cards would be sufficient [Bochkovskiy et al., 2020]. In general, object detectors are described as consisting of a series of components; an input, a backbone, a neck and a head. The input passes information, e.g. an image, to the backbone, which is a pre-trained feature extractor based on a CNN. The neck gathers extracted features from different stages in the backbone. Several neck components can be deployed at once. The head consists of either a two-stage- or one-stage object detector with YOLO categorized as a one-stage detector.

The YOLOv4 model architecture [Bochkovskiy et al., 2020] consists of CSPDarknet53 [Wang et al., 2019] for its backbone, SPP [He et al., 2014] and PAN [Liu et al., 2018] for the neck components and YOLOv3 [Redmon and Farhadi, 2018] as its head one-stage detector, Figure 3. Darknet53 is a CNN backbone intended for use with YOLOv3 [Redmon and Farhadi, 2018] and CSPDarknet53 takes this architecture and applies a CSPNet strategy partitioning the feature layers and increasing the flow rate of computations [Wang et al.,

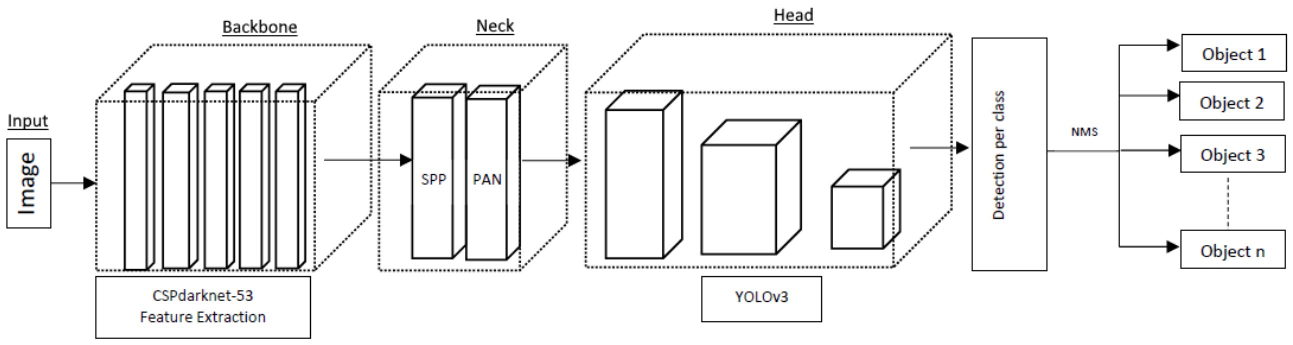

Figure 3: *The general model architecture of the YOLOv4 object detector [Ayoub and Schneider-Kamp, 2021, p. 7]. The backbone consists of CSPDarknet53, the neck of SPP and PAN, and the one-stage detecting head of YOLOv3. Using non-maximum suppression, all detected classes are pruned, and the best candidates for objects are displayed.*

2019]. In the neck Spatial Pyramid Pooling (SPP) allows for variable input sizes to not limit the backbone to a fixed input dimension [He et al., 2014] and Path Aggregation Network (PAN) at its core accelerates information flow between feature layers and boosts prediction inference [Liu et al., 2018]. The head based on YOLOv3 has three detection heads, meaning that YOLO detects objects throughout the processing at three separate feature levels and grid sizes. This efficiently ensures that even small objects are considered during classification and visualize the best candidates using NMS [Redmon and Farhadi, 2018].

The YOLOv4 paper is publicly available, and a Github repository has been purposed for accelerating usage and research into the detector, benefiting the computer vision community at large [Bochkovskiy et al., 2020]. YOLOv4 can run on either processors or graphics cards, and together with a general and at times vague description for custom training, allows for a healthy open-source-based research ecosystem. Training a custom YOLOv4 detector is based on transfer learning utilizing pre-trained weights. Training session- and augmentation parameters of the underlying CNN In the configuration file for the custom YOLO detector can be configured to purpose the detector efficiently. It is recommended that the algorithm runs for at least 6000 epochs and preferably $\#epochs = classes \cdot 2000$. A labeled training and validation data set is required for the algorithm to compare predicted bboxes to properly labeled bboxes using IoU. The network-, data set-, and sample sizes result in high epoch recommendations to train the algorithm within a reasonable timespan using powerful computational hardware.

Evaluating YOLOv4 performance

During training, loss is minimized by calculating the mean square error of IoU on detections in reference to the test data. When training of the detector concludes, YOLOv4 returns a plot of the loss- and mean average precision (MAP) curves. Average precision (AP) is determined for each class by calculating the area under the precision over recall curve (AUC). The curve is drawn continuously using the test data to calculate precision, Equation 1, and recall, Equation 2. Both measures rely on counts of true positive (TP), false positive (FP) and False Negative (FN) detections in relation to the test set, as these are the prerequisites for using the measures. YOLOv4 as a standard uses an IoU of 0.5 correlating to 50% intersection to determine whether a detection should count as either a true- or false positive in relation to the test bboxes. In addition to the IoU threshold, for the detection to be registered as viable it must classify with a confidence of at least 25% or above.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

Given precision and recall, it is possible to calculate the F1 score, also known as the harmonic mean, Equation 3. F1 is an efficient measure to use in distinguishing detectors by their performance.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3}$$

A perfect AP score of 100% would require the algorithm only to detect true positives and no more. What is common however is the algorithm classifying false positives leading to a drop in precision with increasing recall reducing AUC. The average in AP derives from taking the average of precision along 11 points of the curve [0:10,1], leading to the determination of AUC. For multiple classes, MAP is calculated by taking the mean AP across all classes giving the algorithm a collective score on the test set in relation to its efficiency in mastering multi-classification. For individual classes along with AP, counts for true- and false positives and calculated precision, recall, F1-score and average IoU is returned. The standalone measures for classes allow for examining single class performance but also aid in explaining how the MAP score is determined.

Depending on the subject and difficulty level, algorithms performing at an AP above 80% can generally be considered satisfactory good. The cost of misclassification should also be considered when evaluating detectors, as some classes have little real-world costs associated with their misclassification. For example, one would rather misclassify detections for the number of rocks present within aerial imagery than misclassify or miss the presence of endangered plant species. Maximizing the true positive count at the cost of producing false positives can, in instances, be argued to outweigh the costs in loss of precision. The given example would manifest as low precision- and F1 scores but a high recall value.

Configurating YOLOv4

YOLOv4 as default for training utilizes transfer learning for pre-trained weights from the MS COCO data set. Further training on custom classes can be configured in numerous ways. Training time in epochs is dependent on the number of classes, as described previously, and related parameter configurations highly impact the speed of computation. Describing all configuration parameters is pointless as the number of properties is too many to list for this study as over 900 variables are available for tweaking the detector. The parameters deemed most relevant for understanding the usage of YOLOv4 within this study will be described in the following section.

The input size is set for the network in a quadratic layout for which inputs of variable dimensions are scaled to fit by SPP. Batch size is set to feed the algorithm with several images for each epoch, with an additional subdivision being available for graphic cards with low memory capacity. For training the YOLOv4 network, several parameters are available for conventional network setups. The learning rate by default is set to 0.001 and using a scale factor of default 0.1, a step parameter is set at the desired number of epochs where the learning rate is scaled with the factor. This allows the algorithm to dial down the step size in the loss curve gradient descent, allowing for a slow and precise convergence towards the global minima on the loss curve. By default, the scaling factor is activated when reaching the last 20 and 10% of training epochs.

The direct structure of the network layers is accessible to edit but will be refrained from to keep the structure in its recommended and generally optimized state. It is, however, required to indicate the number of classes to train on in each layer and alter the number of filters in the convolutional layers accordingly to the class count, Equation 4.

$$Filters = (classes + 5) \cdot 3 \tag{4}$$

The network can be purposed for detecting small and/or many objects within an image as proposed by YOLOv4 co-creator Alexey Bochkovskiy[Bochkovskiy et al., 2020]. Purposing the detector for many objects within one input image requires adjustment of the last YOLO layer to consider at max 200 -or more objects by adding "$max = 200$" rather than the standard calculated object count derived from the image dimensions, Equation 5.

$$max = 0,06152 \cdot (img\_width \cdot img\_height) \tag{5}$$

To configure for detection of objects smaller than 16x16 pixels after SPP resizing, parameters for number of convolutional layers is dialed down from 54 to 23 and kernel stride doubled from 2 to 4.

Options for data augmentation are built into YOLOv4, and relevant augmentations can be activated and deactivated as seen fit. Data augmentation relates to the possible lack of samples in training the algorithm. Applicable augmentation measures can be implemented to both increase sample size and induce variance in the data set [Connor and Taghi, 2019]. It is a common measure in image classification as slight or significant alterations in duplicate images for the algorithm correlate to being presented with unique and new data never seen before. Augmenting has no substantial downsides if proper methods for the select data set are implemented. Augmentation should be applied cautiously as not all augmentations are representative of select classes, and a certain level of domain knowledge of objects and data types is required when using augmentations.

UAVs photographing nature at nadir can capture multiple objects within a single frame. Through continuous flight on extensive surveys, gathering large amounts of data is of no issue. For detection of plants and inanimate objects such as rocks in the surveyed habitats, being in a top-down view, few restrictions apply as objects are viewed in a two-dimensional space. This eliminates the effects and demands from depth and orientation in properly representing classes within the imagery. A comprehensive suite of augmentations is available in YOLOv4, and those of greatest relevance to habitat surveillance are variables rotation, exposure, HUE, saturation, jitter, random and mosaic augmentation.

At nadir view, objects can freely be rotated 360° and should be encouraged as the data set can be augmented significantly utilizing this one augmentation alone. By default, the rotation parameter is set to 0. Within reasonable levels, exposure, saturation and HUE should be augmented to simulate bright or low light conditions with- and without overhead clouds. The three parameters directly impact pixel values without disrupting the order of values within the image grid. Exposure relates to light balance, saturation to color intensity and HUE deviating from the others and randomly alters color channels rather than setting intensities. HUE can seem counterintuitive but can aid in promoting detector emphasis on learning textures and shapes rather than coloration. By default, exposure is set to ±1.5, HUE to 0.1, and saturation to ±1.5. Jitter randomly crops and resizes imagery with changing aspect ratios within the aspect ratio range from $x(1 - 2 \cdot jitter\_val)$ to $x(1 + 2 \cdot jitter\_val)$ with the jitter value defaulted to 0.3. The Random augmentation is by default active and randomly resizes the network every ten epochs by scales of 0.71-1.4. Mosaic relates to creating so-called mosaics, images overlapping within one another to crop labeled objects and create diverse imagery. Mosaic for UAV imagery is highly applicable as being based on a two-dimensional plane; imagery merging is not too intrusive.

Materials & Methods

Select Danish habitats surveyed via UAV for plant species will be used to train YOLOv4 object detectors. Detectors are constructed to investigate if they yield sufficient value or display promising potential as a substitute to current DEPA habitat surveillance. The general workflow for constructing and evaluating such a detector is dependent on the prior collection of image data from the UAV, Figure 4. The workflow for training detectors will now be provided as a brief overview, with the remaining sections detailing each step of the data-to-product flowchart pipeline in detail.

The image data are split into train, test, and validation sets and labeled manually to give YOLOv4 a reference point to train the detector for, Figure 4I. All data are labeled manually for all classes of interest present within imagery, preferably using a labeling interface like LabelImg, Figure 4II. The train and test splits are used to train the YOLOv4 detector, while the validation set is kept to evaluate detector performance in coverage estimations of objects. YOLOv4 configuration is handled with respect to the given data subject to ensure success-

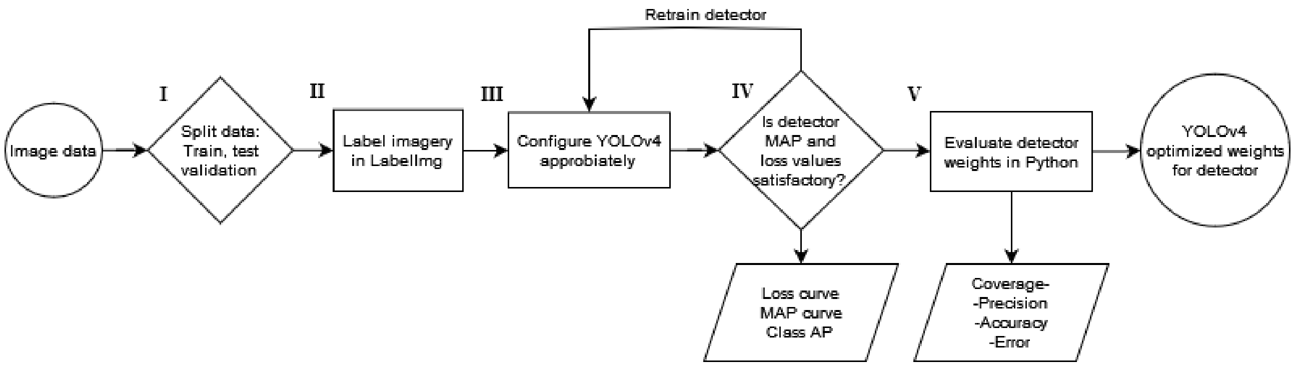
Retrain detector
I
II
III
IV
V
Image data
Split data: Train, test validation
Label imagery in LabelImg
Configure YOLOv4 approbiately
Is detector MAP and loss values satisfactory?
Evaluate detector weights in Python
YOLOv4 optimized weights for detector
Loss curve MAP curve Class AP
Coverage- -Precision -Accuracy -Error

Figure 4: *Workflow for training YOLOv4 object detectors on UAV imagery. Circles symbolize start- and end points, diamonds decisions, squares steps and last trapezes symbolize data outputs. Integral processes are marked by roman numerals in chronological order from start- to endpoint with steps III and IV being the only repeatable steps of the workflow.*

ful training with the highest possible accuracy, Figure 4III. Training of the detector concludes with an output of class AP, calculated MAP, and a graph displaying the loss curve and corresponding MAP value throughout training, Figure 4IV. All measures are calculated for the detector using data given in the test set. If measures like MAP display unsatisfactory results, the detector should be trained again. The final trained detector weights are assessed for accuracy, precision, and error in successfully identifying objects and their corresponding coverage using the validation set, Figure 4V. If found adequate, the detector weights can be deemed as a finished and ready-to-use object detector for the given classes.

This workflow will be repeated numerous times to find the optimal configurations of YOLOv4, determine proper labeling strategies and ensure the best detector for the task at hand is trained and validated. The workflow thus constitutes the empirical process in investigating YOLOv4's capabilities as a UAV imagery object detector.

UAV Platform

This study will utilize a DJI Phantom 4 Pro UAV (P4) as the platform for data collection [DJI, 2016]. P4 is a consumer-grade UAV with standard-issue equipment and a small frame size weighing 1.39kg. P4 is a quadcopter capable of maintaining a steady positioning in several different altitudes, close to the ground and far up in the sky. P4 is additionally equipped with a gimbal for steadiness in capturing imagery and video material with 3-axis stabilization at pitches -90° to +30°. The RGB camera is a 1-inch CMOS sensor with a lens having a field of view of 84° at an 8.8 mm/24 mm f. 2.8–11 aperture. Resolution of imagery is up to 5472x3648pixels for still imagery and 4096×2160pixels for video shot at 30 frames per second. When operating a P4 in Denmark, it categorizes as an A2 platform under the operational category in the weight class C2, UAVs under 4kg [Trafikstyrelsen, 2022]. Operating the platform legally requires the operator to have a drone competency certificate. Batteries for the P4 are of high capacity 4S LiPo (lithium-polymer) at 5350–5870 mAh. These provide a flight time of approximately 22–28min, dependent on the wind conditions while surveying.

Field Protocol and the Lack thereof

Unfortunately, seasonal variance in Danish nature and climate has left little time for capturing meaningful plant life data in this study's duration. In 2022 February had the 2nd most rainfall ever recorded-, [Scharling and Abildgaard, 2022], and March was the driest ever recorded since 1874 by the Danish Meteorological Institute, [Damberg, 2022]. April 2022 has initially, on average, seen colder weather with morning frost and an average morning temperature lower than that for the January and February averages. Growth and blooming worth capturing for a data set were observed around the

14th of April. These extremities in the weather were naturally not foreseen and heavily compromised the study's planned work schedule for collecting data, Appendix A. Due to these unforeseen compromising conditions surrounding the study, it was decided to incorporate old data material gathered by UAV for use in other studies at the start of April.

For the data at disposal, and if the flight was conducted for the study, a series of ground rules are applied to minimize errors related to data quality and handling. As described under "Purposing aerial imagery for image analysis," capturing data midday in mild weather is preferred to minimize the effect of shadows. Steady weather, either overly clouded or no clouds at all, helps lead to uniform capturing of brightness levels. A proper spatial resolution should be calculated beforehand to ensure that objects within the area of interest are appropriately represented visually in the data at a set altitude. Altitudes of flight should be noted to allow other stakeholders than the flight operator to handle the data correctly. If this was not noted, ideally, an object of a known size, e.g. a clipboard that is vibrantly colored, should be provided as a reference, allowing users of the data to calculate the spatial resolution in reference to this object. Through physical ground-truthing, the operator must verify how the terrain should be described, what objects and types of vegetation are present, and estimate the coverage and dispersion of said entities in the area. This is important so as not to misinterpret imagery later to avoid finding objects and relations not present in the survey area. A field protocol detailing the survey area's species composition and physical properties and imagery from the operator's point of view should be sufficient in verifying later observations within UAV imagery.

Data Foundation

Handout data

The study relies on three data sets deriving from a collection of images of three distinct habitat areas:

1. A survey of Daltofterenden, a waterway experiencing heavy growth of the invasive species giant hogweed (*H. mantegazzianum*) along its trajectory while being surrounded by agricultural fields. Daltofterenden is located in the northeastern part of Odense, north of the suburb Agedrup.

2. A survey of Bøjden Nor, a protected bird habitat consisting of beach meadows protected under the Habitat Directive. Bøjden Nor is located in the southwestern part of Fyn, located next to a small rural town and a large ferry berth.

3. A survey of Enebærodde, a protected bird habitat consisting of various habitat types such as beach meadow, juniper bushes, timely wet meadows, wet- and dry heath, and barrier beach with perennial plants. Enebærodde is located on the northern coast of Fyn.

All sets are collected using the P4 UAV platform. Daltofterenden was surveyed at an altitude of 6-10m and a spatial resolution of about 2.1mm per pixel at a resolution of 5472x3648 [Svane et al., 2020, P. 4]. Bøjden Nor was surveyed at 20m altitude with a spatial resolution of 0.55cm per pixel at a resolution of 5472x3648. The data set of Bøjden Nor comprised of only an orthomosaic, a collective merge of all images at 32,913x35,056, and the orthomosaic was cut into images of size 3840x2160 to allow training on the imagery. Enebærodde was surveyed at 100m altitude resulting in a spatial resolution of 2.4cm per pixel at an image resolution of 4864x3648. Niels Svane performed the survey on the first of June 2020. Field protocols were unavailable for the Daltofterenden and Enebærodde surveys. Bøjden Nor was surveyed in July 2021 in relation to a separate study by the author following field protocol guidelines with ground truth'ed presence of sea kale.

Labeling and descriptive guidelines

Captured imagery with little to no overlap is labeled for training using the LabelImg interface for annotating bboxes in imagery purposed

Table 1: *List of data sets used for YOLOv4 models. Sets are provided with IDs. The location for data gathering, a brief description otherwise detailed in the main text, and the subject in focus within each set is listed. The resolution of images in the train, test and validation split is listed along with the number of objects labeled within each set split, respectively (#Train, #Test, #Val). The number of images in each split (#Images) is listed as "Train - Test - Validation". For the set EORS, two validation splits are present; a single image equivalent to the one used for EOR and EORE and seven images being smaller cutouts of the aforementioned image. Due to this, Resolution is marked by an asterisk as validation imagery differs from train and test. Four out of the seven images can be computed, housing an increasing number of objects under #Val.*

ID	Location	Description	Subject	Resolution	#Images	#Train	#Test	#Val
DPS	Daltofterenden	Parameter search	Hogweed sp	5472x3648	5-1-1	470	69	127
BNK1	Bøjden Nor	Three subclasses	Sea kale	3840x2160	8-3-1	862	235	193
BNK2	Bøjden Nor	One main class	Sea kale	3840x2160	8-3-1	862	235	193
BNK3	Bøjden Nor	Flower~Shadow	Sea kale	3840x2160	8-3-1	862	235	193
BNK4	Bøjden Nor	Flower~Kale	Sea kale	3840x2160	8-3-1	862	235	193
BNK5	Bøjden Nor	Kale~Shadow	Sea kale	3840x2160	8-3-1	862	235	193
BNK6	Bøjden Nor	No shadow	Sea kale	3840x2160	8-3-1	579	194	105
BNR1	Bøjden Nor	Refined main class	Sea kale	3840x2160	8-3-1	396	115	78
BNR2	Bøjden Nor	Refined dual class	Sea kale	3840x2160	8-3-1	792	215	158
EOR	Enebærodde	Cluster labeling	Beach rose	4864x3648	10-3-1	254	129	29
EORE	Enebærodde	Segmenting labels	Beach rose	4864x3648	10-3-1	494	162	106
EORS	Enebærodde	Input size matching	Beach rose	*608x608	241-15-1/7	945	61	29/3,11,19,20

for YOLO training [Tzutalin, 2015], Figure 4 II. In LabelImg, bboxes are drawn around objects for each class of interest within every image, Appendix B. Annotating UAV imagery covering large areas naturally leads to multiple instances of classes being present within one image. When drawing bboxes, LabelImg annotates in a separate text file; class number by ID, bbox centroid X and Y location, and bbox width and height.

The process of labeling data is time-consuming and leaves a high rate of error based on the labeling participant's skill set. To reduce error, labeling is performed in a structured manner in general, following a set of ground rules for each data set as different classes of objects can require different methodologies in labeling. No matter the object, this study will strive to strictly enclose objects within bboxes and not leave a large margin of error surrounding the object. YOLOv4 evaluating precision on IoU should naturally only evaluate the object itself rather than the backdrop or shadows surrounding it. Increasing bbox sizes allows the algorithm to draw predictions and receive positive IoU feedback by also grabbing the backdrop, leading to overfitting for the given data set. There will always be an inherent level of error present when labeling data, as the labeling part has a particular bias regarding the identification and interpretation of borders for objects. This cannot be circumvented, but the same operator labels all data to reduce errors otherwise present when multiple participants perform labeling.

Twelve labeled sets are constructed for training YOLOv4 detectors; one set for Daltofterenden, eight for Bøjden Nor, and three for Enebærodde, Table 1. The table in its entirety is available in Appendix C, with the distribution of objects within separate classes in each data set split being listed.

Daltofterenden

One set is created for Daltofterenden, where it is assessed that a population of giant hogweed in different growth states is present. Hogweed leaves are characteristically fringed with spiky leaves varying greatly in size, typically being a lighter green color with almost white veins defining the leaf's structure, Appendix D. Hogweed is an invasive species, and an extensive plan has been outlined for municipalities to combat local populations. The set is named DPS for Daltofterenden Parameter Search as it will be utilized to empirically investigate the impact of YOLOv4 configurations, Table 1 DPS. The set is split into training, test, and validation sets as five, a single, and a single image, respectively, consisting of 470, 69, and 127 objects each. Objects are labeled per leaf and per plant, but differentiating individual plants in heavy growth clusters is very difficult and results in reduced consistency, Appendix E.

Bøjden Nor

Eight sets are created for Bøjden Nor; six for subclass investigations on sea kale, Table 1 BNK1-6, and two refined sets based on the observations made in purposing the prior six, Table 1 BNR1 & 2. The subject is sea kale (*Crambe maritima*) and is treated in labeling with a subclass scheme to allow for exploration of the impact of class definitions. Sea kale is akin to cabbage heads standing with wavy, almost turquoise leaves bearing white flowers on stalks with a light green color before blooming and looks nearly scorched yellow after withering, Appendix D.

Sea kale can be positioned free-standing or within vegetation clusters on a beach meadow. Sea kale is in this study separated into three subclasses; freestanding sea kale, sea kale flowers, and shadowed kale. Sea kale flowers are labeled equally no matter their growth state and recorded as entities within the two other subclasses, Appendix F1 BNK1. Shadowed kale is defined as instances within vegetation clusters or closely related to these at the edges of clusters, Appendix F1 BNK1. Free-standing sea kale is instances standing clearly defined outside vegetation clusters, Appendix F1 BNK1. All sets of Bøjden Nor, BNK, and BNR, consist of the same imagery for the train, test, and validation split at eight, three, and one image, respectively. BNK1 through 6 consists of the same labeled objects, but with class definitions differing between each set, Table 1. BNK6 additionally has the shadowed kale subclass entirely redacted and thus drops in set size but still comprises identical objects to the other BNKs for the remaining classes. 862 objects are labeled for the training set: 197 instances of free-standing kale, 447 flowers, and 218 shadowed kale. For the test set, 235 objects are labeled: 87 instances of free-standing kale, 118 flowers, and 30 shadowed kale. 193 objects are labeled for the validation set: 19 instances of free-standing kale, 86 flowers, and 88 shadowed kale. In descriptions, Table 1 Description, it is observed for BNK2 that all subclasses are merged, BNK3 flowers and shadow kale are merged, BNK4 flowers and free-standing kale are merged, and last in BNK5, free-standing kale and shadowed kale are merged. For BNK6, the class shadowed kale is wholly omitted. The different class definitions are visualized in Appendix F1-2 for BNK1 through 6.

BNR1 and 2 disregard the shadowed kale subclass while refining the labeling process to meticulously label sea kale and sea kale flowers as accurately as possible, Appendix F2. Prior shadowed kale now belongs to the new class of sea kale, including free-standing kale. BNR2 yields the now two subclasses, whereas BNR1 collects the refined labels into one collective sea kale class, Appendix F2. BNR1 consists of 396 training objects, 115 test objects, and 78 validation objects. BNR2 has more objects as flowers are a subset overlapping with the sea kale itself, and for the 792 training, objects are separated into 346 instances of sea kale and 446 flowers. 215 objects are split into 89 sea kale and 126 flower instances for the test set. 158 objects are labeled for the validation set, split into 87 flowers and 71 sea kale.

Enebærodde

Three sets are created for Enebærodde, Table 1., focusing mainly on beach rose (*Rosa rugusa*) EOR and EORE, and one set additionally labeled for lugworm mounds, EORS. EOR and EORE consist of the same images for the training, test and validation set with ten, three and one image respectively. The beach rose grows as a bush and has a lush green color, Appendix D. Being a bush, the rose is often identified as large vegetative clusters and smaller bushes dispersed within the same narrow spot. Labeling of beach roses in EOR is carried out to achieve labeling of uniform clusters of the rose, be it in massive clusters or clusters with varied dispersion, Appendix G. In EORE, labeling is performed with an emphasis on edges, somewhat simulating the effect of segmentation algorithms, Appendix G. In EORE, smaller bboxes are drawn around the edges of vegetative clusters, and the inner parts of the cluster are also segmented into smaller bboxes. This leads to a significant increase in the number of objects to train on. EOR consists of 254 training objects, 129 for the test and 29 in the validation split. EORE consists of 494 training objects, 162 for the test, and 106 objects in the validation list.

EORS differs, consisting of several images at a resolution of 608x608. The set is constructed to investigate claims within the YOLOv4 community stating that training on images equally sized to the SPP neck input can lead to accurate detection of objects in high-resolution images well above the input size; essentially, the goal is to preserve image detail and avoid initial downsizing during training[du Preez, 2019]. Lugworm mounds are included during labeling as these are identified with relative ease given the essentially increased zoom on the low-resolution imagery, Appendix G. Lugworm mounds are visual as darker spots on the seabed, often surrounded by a zone of lighter substrate compared to the surroundings as the worm resuspends sand from its burrow, Appendix D. 241 image cutouts constitute the training set with 450 rose instances and 495 lugworm mounds leading to 945 objects in the training set. For the test set, 15 images with 61 objects are used, of which 20 are rose instances and 41 lugworm mounds.

The single validation image used for EOR and EORE is reused in EORS, as seen by there being also 29 validation objects, Table 1. No lugworm mounds are labeled for the shared 4864x3648 validation image as in testing detectors. We will later see EORS trained models being unable to compute high-resolution images, Table 7. Additional seven images are listed under the validation set, with these being cutouts of the full resolution validation image shared across the EO sets. The cutouts are of size 608x608 and are in sequence arranged in grids to investigate the detector's performance with increasing image resolution. Cutout grids are arranged sequentially as 1x1, 2x2, 3x3, 4x4, 5x5, 6x6 and 7x6. Grid cutout 8x6 is equivalent to the original full-size image at 4864x3648. As for the full-size validation image, 5x5, 6x6 and 7x6 will prove to be unable for computation, Table 8., and no object count is given for these in Table 1.

Constructing the Detector

Technical setup

YOLOv4, as proposed and distributed by Bochkovskiy, Wang and Liao [Bochkovskiy et al., 2020] will be used for training custom classifiers utilizing pretrained YOLOv4 weights on the MS COCO data set. Training of the model is performed on an external research system accessed via an SSH connection during the project. The research setup is an Ubuntu 20.04.4 system with 256GB RAM, four Nvidia GTX 1080 TI 10GB VRAM graphics processors, and an eight-core Intel Xeon E5-2620 v4 central processing unit.

Using Docker[Merkel, 2014], an Nvidia image package purposed for CUDA acceleration is deployed to train the algorithm on the external research system using CUDA 11.3.0 and the deep learning module CUDNN 8.2.0.53 [NVIDIA, 2022a]. Upon deploying the Docker image, OpenCV 4.2.0[Bradski, 2000] is loaded in as it is a prerequisite for using YOLOv4. The Docker workspace image is deployed using only one GTX 1080 TI in the host system to allow for up to four docker images running simultaneously. Darknet is compiled with OpenCV and CUDA reliance for Pascal graphics cards like the GTX 1080 TI model. Pascal models do not possess tensor cores which significantly speed up deep learning tasks; this should be kept in mind when examining the time costs during training.

Table 2: *Fixed and variable parameters with value ranges for YOLOv4 configuration in training detectors on the DPS, Daltofterenden Parameter Search, data set.*

	Exposure	Saturation	Hue	Jitter	Random	Input dim.	Batch size	Subdivision	Rotation	Mosaic	Many obj.	Small obj.
Variable	N	N	N	N	N	Y	N	Y	Y	Y	Y	Y
Value	±5%	±10%	10	0.3	On	416-800	64	16/32/64	±0/180	On/off	On/off	On/off

Table 3: *Fixed and variable parameters with value ranges for YOLOv4 configuration in training detectors on the BNK and BNR data sets of Bøjden Nor.*

	Exposure	Saturation	Hue	Jitter	Random	Input dim.	Batch size	Subdivision	Rotation	Mosaic	Many obj.	Small obj.
Variable	N	N	N	N	N	Y	N	Y	N	N	N	N
Value	±5%	±10%	10	0.3	On	416-800	64	16/32/64	±180	On	Off	Off

Table 4: *Fixed and variable parameters with value ranges for YOLOv4 configuration in training detectors on the EO data sets of Enebærodde.*

	Exposure	Saturation	Hue	Jitter	Random	Input dim.	Batch size	Subdivision	Rotation	Mosaic	Many obj.	Small obj.
Variable	N	N	N	N	N	Y	N	Y	N	N	N	N
Value	±5%	±10%	10	0.3	On	608-800	64	16/32/64	±180	On	Off	Off

YOLOv4 model configurations

YOLOv4 is configured in different ways depending on the type of data for the given detector, Figure 4 III. For the DPS data, YOLOv4 is investigated empirically for parameter influence on trained detector performance, Table 2. The pixel modifiers exposure, saturation and HUE are set to fixed custom values across all training sessions, Table 2. These custom values are deemed to stay true to the representation of objects in nature with exposure set to 5%, saturation 10% and HUE to 10, Appendix H. Augmentations jitter and random network resizing are set to default active values remaining static throughout training, Table 2. Augmentations rotation and mosaic are variable between sessions, Table 2. Network-structure related parameter configurations for small- and many objects are variable together with PAN input resolution size for which batch subdivisions are adjusted accordingly to accommodate vRAM capacities, Table 2.

Using the BNK and BNR data, labeling strategies are investigated to determine whether the detector benefits from training on distinct subclass definitions or benefits from single class labels per object. Unlike DPS sessions, BNK and BNR utilize the same configuration throughout training sessions. Only configuration for input size and declared number of classes vary with the latter being adjusted for in network filters, Equation 4, Table 3. BNK and BNR are being investigated after DPS, and static parameters are thus set to values deemed reasonable while training detectors for DPS, Table 3. Dependent on the input size, the subdivision is also tweaked accordingly to prevent vRAM overloading. As for BNK and BNR, the EO sets are configured in a fixed manner relying on observations for DPS, leaving only input size and subdivision as variable, Table 4. EORS having two classes present is configured for this in the configuration file as is required to initialize training, Equation 4.

Accuracy assessment

As provided by the algorithm, trained detectors are evaluated on a per-class level with AP and the entire detector on MAP level for the test set, Figure 4 IV. Training curves are plotted with visualized MAP and training loss. Loss is visualized across all epochs of training, while MAP is only visualized at epoch 1000 and every subsequent 100th epoch following. Recall, precision and F1 are calculated in the final epoch for the final detector weights based on the test set. The measures are reliant on the IoU threshold at 0.5 to be met between the detection bboxes and test ground-truthing bboxes for them to score as TP. Higher scores are thus reliant on detections being spatially accurate for every single bbox instance, leading to the evaluation emphasizing accuracy and spatial distribution.

With multiple objects being present within

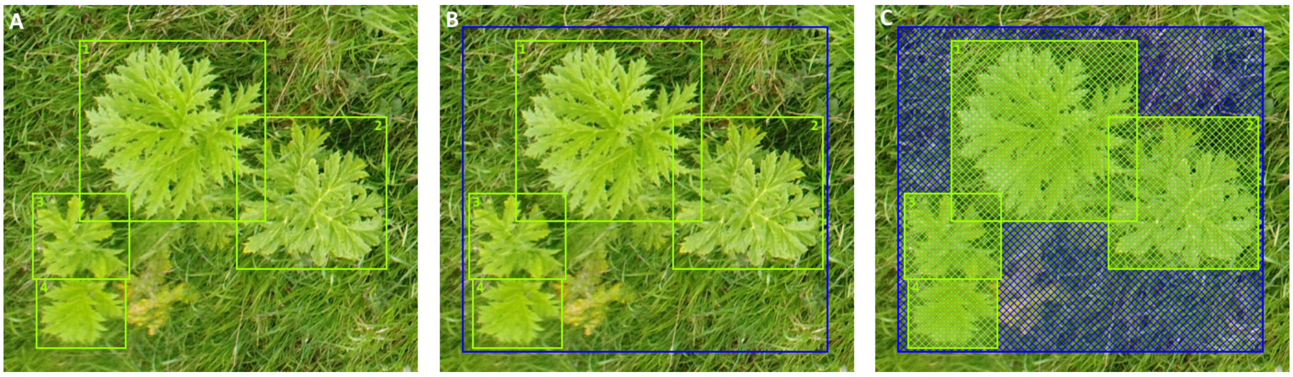

Figure 5: *Labeling of hogweed with four yellow bounding boxes, A, in relation to a blue YOLO imitated bounding box surrounding all labeled instances, B. C sees bounding boxes filled to give a comparative outlook on how the relation between bounding box coverage is between labels and the single detection.*

the validation images, 29-193 Table 1. #Val, overlap between objects is common, Figure 5A. The detector risk drawing bboxes around multiple objects that subsequently score negatively as FP when they fall below the IoU threshold for the individual objects, Figure 5B. These FPs can still be considered accurate detections below the IoU threshold as they successfully encircle objects; however not at the desired IoU threshold. To investigate this aspect further, accuracy measures are repurposed to evaluate the accuracy, precision, and error rate as under- and overestimation based on IoU coverage between the validation set and detectors predicted bboxes, Figure 5C, using a separate python script, Figure 4 V.

Measures now emphasize positive feedback through accurately drawing bboxes around coverages even if the detectors bbox differs spatially and strategically from the labeling parts validation bbox. As objects often are significantly smaller than a meter in diameter, scoring negatively for detections under the ground-truthing IoU threshold, 0.5 is arguably counter-intuitive as the important factor is that the object is detected. One meter of inaccuracy in the detection is not essential when surveying habitats spanning hectares by UAV. High coverage precision will convey that the detector spatially is accurate but not necessarily with respect to every single object and bbox, but rather the population of the class within the image. Detectors now score positively despite drawing bboxes differing from the strategy defined in the validation set, and the model is no longer penalized for finding alternative yet accurate detection strategies. Any bias towards the applied labeling strategy is thus diminished during detector validation.

The coverage evaluation script is written in Python 3.8.12 with dependencies on OpenCV 4.5.5.62, Pandas 1.4.2, Numpy 1.22.1, Shapely 1.8.2 and Matplotlib 3.5.1. Provided the detector .cfg- , .weights- and .names file (configuration, weights, class ID) the YOLOv4 detector is quickly compiled using Darknet via OpenCV. Given an image, related validation bbox coordinates, and class labels, the script uses the mounted detector to detect objects within imagery on a per-class basis. The detector creates a binary validation- and detection mask for each class. Masks are constructed by a collective of single standing bboxes and merges of overlapping bboxes. Masks are bitwise images where bbox rectangles are filled and colored black for value 1 and the remainder of the picture colored white for value 0. These masks are used to evaluate the detector's performance through bitwise comparisons.

Coverage accuracy (CA) is evaluated taking the fraction, of detection and validation IoU, relative to validation coverage, i.e. CA is the fraction of the validation covered by the IoU, Equation 6:

$$CA = \frac{IoU}{Validation} \tag{6}$$

Coverage underestimation (CU) is simply CA subtracted from 100, resulting in the fraction of the validation not covered by the detector, Equation 7:

$$CU = 100 - CA \tag{7}$$

Coverage precision (CP) is the fraction of, detection and validation IoU, relative to summed validation coverage and the IoU subtracted from the detection coverage, i.e. CP is IoU coverage relative to the collective coverage of the detection and validation mask, Equation 8:

$$CP = \frac{IoU}{Validation + (Detection - IoU))} \tag{8}$$

Coverage overestimation (CO) is the fraction of, detection and validation IoU, relative to detection coverage, subtracted from 100 i.e. CO is the fraction of the detection that is not part of the IoU, Equation 9:

$$CO = 100 - \frac{IoU}{Detection} \tag{9}$$

The F1 metric remains and is calculated for the validation based on TP, FP, and FN with the IoU threshold at 0.5 as is the standard, Equation 3. F1 conveys both detector performance and the accuracy of the spatial distribution of objects, proving the traditional YOLOv4 metric remains a relevant supplement to the coverage evaluations.

The evaluation script outputs measures and imagery on a per-class basis. Before constructing the masks, the overlap between detections and validation bboxes at an adjustable IoU threshold, standard 0.5, is utilized to count TP, FP, and FN. These counts and subsequent calculation of F1 are dependent on Shapely identifying and counting bbox rectangle overlaps for set IoU. TP, FP, and FN are determined by relying on a modified code courtesy of K.E. Koech, [Koech, 2020]. The script presents counts and the F1 score as text outputs, Appendix I1. Mask coverage of classes, provided input of the image's spatial resolution, in actual world dimension is returned for the IoU-, detection-, validation-, over-and undershoot coverage as text. Being reliant on the aforementioned, CA, CU, CP and CO are likewise calculated using the masks and outputted in text. The IoU mask and differences between the validation and detection masks are visualized as CA, CO and CU atop the input image, Appendix I1. A sample of the collective output of the evaluation script can be seen in Appendix I1 and I2.

For the highlighted example in Figure 5C, the new metrics would result in a CA at 100% as all labeled instances are totally enclosed. No validation bboxes stretch- or are positioned outside the detection, and CU would thus be 0%. CO would approximately be around 40% as the detector overestimates coverage significantly, leading to CP clocking in at around 70%.

Table 5: *YOLOv4 detectors trained on detection of Heracleum sp., hogweed. Configuration of YOLOv4 parameter settings is listed. Rotation refers to enabling ±180°image rotation, whereas Many Obj and Small Obj refers to editing the YOLOv4 configuration file to optimize detection for many and/or small objects. Mean average precision at 50% overlap (MAP50) is calculated on the test set and listed for the weights at the conclusion of training. With only one class in the data set, MAP50 is equivalent to average precision. Coverage accuracy (CA), coverage underestimate (CU), coverage precision (CP), coverage overestimate (CO), and F1 score is calculated on the validation set with IoU and NMS threshold set to 0.5 at a confidence level of 0.8.*

Model	Input dim.	Subdiv.	Rotation	Mosaic	Many Obj	Small Obj	MAP_{50}	CA	CU	CP	CO	F1
DPS0	416	16	N	N	N	N	20.05	94.2	5.8	87.3	7.8	0.956
DPS1	800	64	Y	N	N	N	20.86	94.9	5.1	87.3	8.5	0.955
DPS2	608	32	Y	N	N	N	17.8	94.7	5.3	87.9	7.5	0.956
DPS3	416	16	Y	N	N	N	23.63	94.4	5.6	87.6	7.6	0.956
DPS4	416	64	Y	N	N	N	20.11	94.4	5.6	87.6	7.6	0.956
DPS5	416	16	Y	Y	N	N	27.03	45.8	54.2	43.7	9.3	0.414
DPS6	416	16	N	Y	N	N	10.69	94.3	5.7	87.6	7.6	0.952
DPS7	416	16	Y	N	Y	N	11.35	94.4	5.6	87.3	8	0.952
DPS8	416	16	Y	N	Y	Y	11.34	94.5	5.5	87.3	8	0.956
DPS9	416	16	Y	N	N	Y	6.35	94.7	5.3	87.1	8.4	0.956
DPS5O	800	64	Y	Y	N	N	29.01	94.1	5.9	87.7	7.2	0.956

Table 6: *YOLOv4 detectors trained on detection of Crambe maritima, sea kale. Configuration of YOLOv4 parameter settings is listed. Mean average precision at 50% overlap (MAP50) is calculated on the test set and listed for the weights at the conclusion of training for classes sea kale(AP K), sea kale flowers (AP F), shadowed kale (AP S), and combinations thereof (AP C) if present. Coverage accuracy (CA), coverage underestimate (CU), coverage precision (CP), coverage overestimate (CO), and F1 score is calculated on the validation set with IoU and NMS threshold set to 0.5 at a confidence level of 0.8. The listed measures are averages across all classes present as calculated during validation. "NA" is omitted entries, and "-" marks entries for classes part of the present combination class.*

Model	Input dim.	Subdivision	MAP_{50}	AP_k	AP_f	AP_s	AP_c	CA	CU	CP	CO	F1
BNK1	416	16	23.31	41.73	26.25	1.94	NA	18.9	81.1	12.0	32.4	0.194
BNK2	416	16	30.44	-	-	-	30.44	28.5	71.5	26.8	17.7	0.304
BNK3	416	16	24.64	34.18	-	-	15.1	27.8	72.3	16.9	41.2	0.309
BNK4	416	16	16.28	-	-	1.68	30.87	11.8	88.3	11.1	20.3	0.158
BNK5	416	16	25.3	-	17.17	-	33.44	11.9	88.2	11.1	34.1	0.129
BNK6	416	16	27.86	42.61	13.1	NA	NA	35.6	64.4	20.3	46.1	0.451
BNK6O	800	64	33.77	40.93	26.6	NA	NA	26.6	73.5	22.1	24.1	0.157
BNR1	416	32	37.02	-	-	NA	37.02	56.6	43.4	52.8	11.3	0.750
BNR1O	800	64	41.74	-	-	NA	41.74	46.4	53.6	43.6	12.2	0.595
BNR2	416	32	24.88	35.79	13.97	NA	NA	30.2	69.9	28.1	18.3	0.367

Results

Daltofterenden, Configuration Influence

Utilizing the DPS data, YOLOv4 performance was explored for several augmentation parameters and structural CNN edits, Table 5. Eleven detectors are trained, all differing from one another in configurations in accordance with Table 2. but all remain dependent on the DPS data set, Table 1. All detectors are evaluated using their final weights file. MAP is determined in training at IoU 0.5. Training graphs with visualized MAP- and loss curves are presented in Appendix J1-2, and the final MAP value is listed in Table 5. CA, CU, CP, CO, and F1 are determined in the evaluation script with the NMS and IoU threshold set to 0.5 and the minimum confidence level set to 0.8. DPS5O is an "optimized" version of DPS5 with increased input size and adjusted subdivisions to prevent vRAM overload. DPS5 was chosen for optimization as it displayed the highest MAP. DPS5O is trained as a triplicate batch and DPS0, 3 and 6 as duplicate batches. For triplicates and duplicates, the detector weights with the highest MAP are presented, Table 5. Fifteen detectors have been trained on DPS, collectively requiring 412 hours of computation for training. All detectors display high CA, CU, CP, CO and F1 measures except DPS5, instead displaying the highest MAP of the not-optimized detectors.

Bøjden Nor, Class Definitions

Utilizing the BNK and BNR data, Table 1., YOLOv4 performance was explored for different interpretations of class definition across three subclasses of the superclass sea kale, Table 6. Model ID is directly equivalent to the ID of data sets used, Table 1., with the "O" models BNK6O and BNR1O being optimized versions of their counterparts displaying the highest MAP. BNK6 MAP falls below that of BNK2 yet was chosen for optimization. The reasoning for this decision is that BNK2 includes the subclass shadowed kale displaying low AP, and its omission in BNK6 was preferred. Ten sea kale detectors are trained and presented in Table 6. MAP and AP are determined in training at IoU 0.5. Training graphs with visualized MAP- and loss curves are presented in Appendix L1-2, and the final MAP value is listed in Table 6. AP is presented for each of the three subclasses, and combinations of these if present.

For models, "-" entries indicate classes incorporated into the present class combination, and "NA" indicates classes omitted in the given model. CA, CU, CP, CO and F1 are determined in the evaluation script with the NMS and IoU threshold set to 0.5 and minimum confidence level set to 0.8. CA, CU, CP, CO , and F1 are average values for the present classes. Coverage measures for every class individually can be seen in Appendix K. BNK2 is trained as a duplicate batch. BNK6, BNK6O, BNR1 and BNR1O are trained as triplicate batches. For duplicates and triplicates, the model with the highest MAP is presented. Nineteen detectors have been trained collectively, requiring 721 hours of computation for training.

Enebærodde, Labeling Strategy and Resolution

Utilizing the EO data, Table 1., YOLOv4 performance was explored for two different labeling strategies; ordinary object labeling, EOR, and labeling objects in segments, EORE, Table 7. The EORS detector differs in being trained on 608x608 cutouts and having a similar labeling strategy to EOR. Model ID is directly equivalent to the ID of data sets used, Table 1. All models were trained in triplicate batches, and the model with the highest MAP was presented, Table 7. Nine detectors have been trained collectively, requiring 298 hours of computation for training. EORS is the only model trained for the lugworm mound class, and '-' for EOR and EORE indicates that the class is omitted. MAP and AP are determined in training at IoU 0.5. Training graphs with visualized MAP- and loss curves are presented in Appendix M, and the final MAP value is listed in Table 7. CA, CU, CP, CO and F1 are determined in the evaluation script with the NMS and IoU threshold set to 0.5 and minimum confidence level set to 0.8. CA, CU, CP, CO , and F1 are average values for the present classes. EORS has multiple validation images associated with the set, Table 1, and is in Table 7. evaluated for the same full-size image as used in evaluating EOR. "NA" indicates for EORS that the model could not detect any objects within imagery, leading to no measures.

The EORS model is evaluated on smaller validation images in grids, Table 8., being small

Table 7: *YOLOv4 detectors trained on the detection of Rosa rugosa, beach rose, and lugworm hills. Configuration of YOLOv4 parameter settings is listed. Mean average precision at 50% overlap (MAP50) is calculated on the test set and listed for the weights at the conclusion of training for classes beach roses (AP Rose) and lugworm mound (AP worms). The "lugworm mound" class is only trained for EORS. Coverage accuracy (CA), coverage underestimate (CU) coverage precision (CP), coverage overestimate (CO) and F1 score is calculated on the validation set with IoU and NMS threshold set to 0.5 at a confidence level of 0.8. The listed measures are averages across all classes present as calculated during validation. "NA" are entries that are not available to be calculated, and "-" marks classes omitted for the given model.*

Model	Input dim.	Subdivision	MAP_{50}	AP_{Rose}	AP_{Worms}	CA	CU	CP	CO	F1
EORS	608	32	75.24	94.78	55.7	NA	NA	NA	NA	NA
EOR	800	64	16.5	16.5	-	89	11	83	7.5	0.642
EORE	800	64	17.57	17.57	-	87.5	12.5	70.5	21.6	0.511

Table 8: *Evaluation of the YOLOv4 trained EORS detector using measures coverage accuracy (CA), coverage underestimate (CU), coverage precision (CP), coverage overestimate (CO) and F1 score for class beach rose within validation imagery with IoU and NMS threshold set to 0.5 at a confidence level of 0.8. Validation imagery is grid-based, with every tile being of the resolution 608x608 and all cutouts of the same image at native 4864x3648. "NA" mark measures on imagery where the detector cannot detect objects of the subject class.*

Image grid	Validation Resolution	CA	CU	CP	CO	F1
1x1	608x608	96.2	3.8	54.8	44	1
2x2	1216x1216	92.3	7.7	71.6	23.9	0.5
3x3	1824x1824	88.3	11.7	74.5	17.3	0.25
4x4	2432x2432	92.3	7.7	70.6	25.1	0.33
5x5	3040x3040	NA	NA	NA	NA	NA
6x6	3648x3648	NA	NA	NA	NA	NA
7x6	4256x3648	NA	NA	NA	NA	NA

cutouts of the common validation image shared by all EO models, Table 7. EORS is trained to detect beach roses and lugworm mounds, Table 7. CA, CU, CP, CO and F1 is evaluated with both NMS and IoU threshold set to 0.5 and confidence level 0.8 in the evaluation script. Measures are presented as averages, but grids 5x5, 6x6 and 7x6 are the only images presenting lugworm mounds. "NA" indicates for imagery that EORS is unable to compute and find any objects of classes beach rose and lugworm mound within the validation imagery. All measures in Table 8. therefore only represent the class beach rose.

Discussion

Implementation, Results and Relations

Before discussing results, one thing should be made clear; utilizing three data sets slightly undermines the intended empirical process. One can call the investigation empirical within each set but not between them. This does not invalidate findings and comparisons but should be considered when drawing relations between detectors with different data set dependencies. The decision to use multiple data sets was prompted by the described time constraints imposed by weather and climate.

Another aspect to note is that detectors' performance was initially tuned and evaluated solely on AP and MAP. Measures CA, CU, CP and CO were proposed and evaluated on the validation set following the conclusion of training of all detectors. Many decisions in designing detectors and tuning their configuration are thus reflective of their MAP score.

Daltofterenden, hyper parameter search

Parameter configuration for DPS displays adverse effects of parameters across trained detectors, Table 5. DPS1, 2 and 3 investigated the impact of SPP input dimensions and led to a discovery opposing what would be the expected outcome. With increasing input size, the input imagery is not downsized as extensively. MAP counter-intuitively drops with increasing input dimensions, Table 5, and time consumption of training rises significantly from 12, 20 to 45 hours for input dimensions 416, 608 to 800 respectively. DPS0 can be considered a blank baseline, and even it displays higher MAP than DPS1 and 2. Comparing DPS0 and 3, it is evident that the rotation augmentation at ±180°leads to increased MAP. Additionally, the rotation augmentation leads to no increase in time consumption for training. The mosaic augmentation tripled the time consumption for training between DPS0 and DPS6 from 10 to 30 hours. DPS6 was run as a duplicate, with both displaying low MAP at 10.7% compared to DPS0, achieving 20.05%. Applying rotation augmentation along mosaic in DPS5 boosts MAP significantly to 27%, revealing DPS5 as the best detector when focusing on MAP, Table 5. In investigating the configuration for small and many objects, DPS7, 8 and 9, mosaic augmentation was turned off to decrease the time consumption of training. Subsequently, DPS7, 8 and 9 are comparable to DPS3.

Editing the configuration to consider many objects, DPS7, had no impact on the training time but dropped MAP significantly by 12% compared to DPS3. Editing the CNN to be inclusive for smaller objects, DPS9, led to another significant drop in MAP with computation time tripling compared to DPS3. Activating both configurations in DPS8 led to low performance at high training consumption. DPS4 was a model identical to DPS3, with the only difference being a batch subdivision set to 64 rather than 16, which displayed a slight drop in the MAP by 3%. DPS4 was run as the only detector with the given configuration, and one could be tempted to state that the subdivision directly impacted the detector's performance. Without a triplicate, this might as well have been a coincidence between the detector's random states.

Configuring for rotation and mosaic augmentation increased MAP scores in the DPS5 model at 23.6%. This resulted in DPS5 being chosen as the optimal baseline parameters for hogweed detection. Despite DPS1 at high input dimensions displaying low MAP, an increased input size at 800 was incorporated for DPS5O to promote accuracy. For the DPS5O triplicates, MAP peaked at 29%, triumphing over DPS5 as the best detector for hogweed. CP for all detectors is satisfactory high at >85% and F1 > 0.95 except for DPS5. This seems counterintuitive as DPS5 was chosen as the basis for the optimized detector. Examining the loss curve for DPS5, Appendix J2, loss was not minimized successfully. DPS5's high MAP is perhaps a result of the detector being lucky in the test evaluation. However, luck does not translate to CP as it might result from having fitted to the background around subjects rather than the subjects themselves. DPS5 has the worst CP at 43.7%, grossly underestimating CU 54.2%. DPS5O manages to achieve the second-highest CP at 87.7%, with DPS2 triumphing at CP 87.9%.

Based on these observations, optimizing DPS2 by including mosaic augmentation would be interesting, as this saw a significant effect in DPS5. Before doing this, it would be wise to run DPS5 as a triplicate to investigate whether the insufficient minimization of loss could be avoided and lead to either higher or lower MAP and CP scores. With CP >85%, the hogweed detectors are deemed satisfactory in their precision. Having F1 scores at 0.95 and above, the detector is spatially accurate, and good localization coupled with high CP makes it a robust series of detectors. In conclusion, there should be little in the way of encouraging the deployment of the detector in surveillance of the invasive giant hogweed.

Bøjden Nor, class definition

Detectors for sea kale fared less successfully than the hogweed detectors, as is evident from the MAP and CP values, Table 6. BNK1 with all subclasses of kale immediately reveals that the subclass shadowed kale is extremely weak at AP 1.94%, as is observed in BNK4 at AP 1.68%. Free-standing sea kale fares well at AP 41.73% and flowers at a similar level to hogweed at AP 26.25% in BNK1. This is also reflected in CP scores across all detectors, where the presence of shadowed kale significantly diminishes the average CP score for BNK1 through 5, Appendix K. Omitting shadowed kale and transforming its labels to regular sea kale leads to better performance for detector BNK6. This is no surprise as the only difference between the free-standing and shadowed kale is whether the object is associated with a vegetative cluster. Combining shadowed kale with one other subclass in BNK3 and 5 decreases MAP and CP performance. Through the subclassing scheme, it is evident that shadowed kale is no success.

Interesting is the assessment of sea kale as one superclass with BNK2 achieving an AP of 30.44% and CP at 26.8%, Table 6. As described, the presence of shadowed kale within BNK2 was deemed undesirable, and hence optimization focused on BNK6, which displayed MAP at 27.96%. In retrospect, this decision was biased and should not have been the determining factor for optimization, MAP should have been the determining factor. BNK6O outperforms BNK6 and BNK2 in MAP score, but BNK2 remains the best detector outperforming the others with CP 26.8% rather than the "optimized" 22.1%, Table 6.

Looking at the BNR models, CP increases

significantly when omitting shadowed kale and refining bbox ground-truthing. BNR2 is comparable to BNK6 and sees decreased MAP and AP scores for the collective kale and flower class but an increase in CP at 6%. The improvement is observed with one refined superclass for sea kale, with MAP scoring 37% for BNR1 and 41.74% for the subsequent optimized version BNR1O. Interestingly, CP for BNR1 succeeds BNR1O, scoring 52.8%, with the latter only achieving 43.6%. It is likely that BNR1O suffers from an increased input size and related decreased subdivision, as discussed in the comparison between DPS5O, DPS5 and DPS2. Both BNR1 and BNR1O display the most stable and minimized loss curve, with MAP retaining its high value throughout training, Appendix L2.

Seeing models with one superclass for sea kale prevailing, it is safe to conclude that creating a subclass scheme for the class sea kale yielded no value. For the best detector, BNR1 with CP at 52.8% is deemed insufficient, and deployment of the detector is discouraged. BNR1 has a CU at 43.4% and CO at 11.3%. Missing many instances of kale and subsequently missing out on extensive coverages is not acceptable for the detector. With CP falling well below the desired 80% CP threshold, the use of BNR1 and remaining weaker detectors cannot be recommended.

Enebærodde, labeling strategy and input size

Enebærodde was the final data set used to construct detectors. EO sets build on previous findings and investigates improvements hypothesized to be interesting while constructing and evaluating the hogweed and sea kale detectors. EOR and EORE are the most similar detectors, both being trained on high-resolution imagery. EOR is constructed using best practices from previous experiences with DPS and BN models by performing gross labeling of objects no matter their size or subdivisions within tight clusters. EORE was constructed with an emphasis on labeling data on mimicking the effect of segmentation by drawing smaller bboxes constituting larger objects. This was decided as the hypothesis of the detectors being penalized by YOLOv4's focus on IoU adjustment during training had been raised, Figure 5. Training the detector to focus on smaller segments could lead to the detector drawing smaller and more accurate bboxes with minor overestimation around the objects.

Results in Table 7 display similar MAP values for EOR and EORE, but with EOR achieving a CP at 83%. EORE only achieved CP 70.5% with three times as high CO at 21.6% yet with an almost identical CA to EOR at 87.5%. Recall that labeling of the validation set for EORE utilized small segmenting bboxes as well, and having a look at the detections in relation to these could explain the tripled CO. Appendix N presents detections on a cutout of the validation set for EOR and EORE. Interestingly the gross labeling strategy from EOR has been adapted by EORE even though it was provided small bboxes for its training. This seemingly is the reason for observing high CO for EORE. EORE is most likely just as precise as EOR if one were to try out both validation images on EOR and EORE, respectively. YOLOv4 displays a clear preference for gross labeling objects even if it is instructed in training to train for smaller objects. An impressive feat highlighting how YOLOv4 remains accurate and accessible for users as even "bad labeling" can lead to accurate models.

Honing in on EORS, we see MAP reaching a level of 75.24% surpassing all detectors trained in the study, Table 5-7. AP for lugworm mounds is high at 55.7% but pales in comparison to the class beach rose at 94.78%. If lugworm mounds were omitted and training initialized anew, the MAP curve would rise to levels of 95% with controlled minimization of loss, Appendix M. Therefore, the results for EORS are more in line with what we observe in the literature, achieving minimized loss and very high MAP. Recall that EORE is trained on images of 608x608 equivalent to the input size of the detector. The input is simpler with roughly 370,000 pixels compared to the 800x800 detectors at 640,000 pixels that have additionally been downsized more than 24 times upon input. This leads to EORS being incapable of detecting objects at higher resolutions, with 2342x2342 being the threshold,

Table 8. EORS fails to infer the validation set at its native resolution, Table 7. Inspecting the detections on variable validation resolutions, Appendix O reveals EORS is struggling in detecting small objects. Only large objects are seemingly detected, leading to a high CP and CA as the detector catches the majority of objects, Table 8. F1 in turn suffers as smaller objects count towards FN and EORS spatial accuracy rapidly diminishes. With small imagery, YOLOv4 seemingly has little trouble detecting beach roses, Appendix O. The MAP curve naturally conveys and measures this efficiency when evaluating performance on the test set as it evaluates other small image cutouts, Appendix M. The fact that the imagery houses so few and seemingly easily identifiable objects diminish the effect of differing label strategies, Figure 5., as little room is left for the detector to interpret objects differently, Appendix O.

Even though MAP alone would lead one to believe EORS is the best detector, feeding larger input imagery and examining CP and F1 reveals it being an unsuccessful endeavor, Table 7 & 8. For surveys with UAVs, where their high spatial- and image resolution is their specialty, having detectors unable to utilize this aspect is simply unacceptable. EOR and EORE are considered equal in their performance but relying on Table 7, the recommended detector for beach rose detection in high-resolution imagery is EOR with a CP of 83%. With a CA of 89% and a CU and CO <11%, the low F1 score of 0.64 should not discourage the use of the detector in the fight against the invasive beach rose.

Evaluation

Empirical process

As highlighted going into the discussion, the study does not follow a strictly empirical process. Rather than building upon the same baseline, three data sets are utilized to investigate different aspects of purposing detectors, drawing from each other's findings. Ideally, the same data set would be used throughout the process, but yielding different properties (e.g. subject type/class) incorporating multiple sets allowed further inquiries due to the increased sample size and variation. An example is how beach rose was the only class subject for which detections varied across very large objects to smaller cluster populations.

Each data set separately consists of imagery at only one distinct altitude resulting in three spatial resolutions of 2.1mm, 0.55cm, and 2.4cm per pixel. Ideally, the effect of different altitudes would, and could in future studies, be investigated with respect to a single subject class. It would be interesting to investigate; 1. What altitudes, and resulting spatial resolutions, are applicable for accurate detection of a subject using YOLOv4, 2. can detectors trained in one altitude remain accurate when tested on other altitudes, and 3. is there perhaps a benefit to training detectors on a mix of different altitudes? Increasing altitude results in greater coverage at the cost of spatial resolution, and this is one aspect lacking in the investigation for detector optimization. The same goes for image resolution, where an investigation into the interaction between variable image resolutions at a set spatial resolution could affect detector performance as the input is downsized. It is not reasonable to conclude that detectors are in their optimized state, given so many relevant inquiries are left unanswered. Further performance gains seem likely given more research into an empirical process emphasizing image- and spatial resolutions.

Labeling objects

It is difficult and time-consuming to label objects for data sets. UAV imagery taken at nadir is challenging to interpret during labeling as lacking the third dimension of height makes distinguishing classes from the remaining terrain and objects difficult, Appendix D. Having the data not being gathered by the author, and in Bøjden Nor's case being collected a long time ago, no real reference is provided to distinguish subject classes from remaining objects. With no sense of object height and image depth, trees suddenly look like a bush within imagery as the tree stump is not visible and canopies, roughly speaking, are bushes atop a pillar. The view at nadir within imagery differs from the frontal view of ordinary imagery, where the height and depth are preserved. Many YOLO studies do not incorporate imagery shot at nadir, and it is difficult to compare findings to subjects in literature. However, what is certain was the successful detection of hogweed and beach roses in validating detectors. Images from Bøjden Nor of sea kale were not successfully mastered by constructed detectors and were unable to be validated sufficiently. Sea kale is easily identifiable *in situ* but on nadir imagery, difficult as textures and sometimes coloration of gravel, rocks, and vegetation mosaics are similar to the subject, Appendix D. Either labeling was insufficient, or the image was very intricate and difficult for the algorithm to infer successfully. Examining loss curves for BNO and BNR models sees loss curves being insufficiently minimized with only the best detectors, BNR1 and BNR1O nearing reasonable levels, supporting the hypothesis of inadequate labeling or data set intricacy, Appendix L2.

Training and evaluating detectors

An emphasis on IoU in validating detectors was deemed "inappropriate" in evaluating detections as clustered detections could be penalized for drawing large bboxes on multiple objects, Figure 5. This related to the detector adopting a different method for drawing detections than that of the labeling participant, resulting in a negative evaluation. YOLOv4 minimizing loss by calculating the mean squared error of IoU during training faces the same inherent problem. Throughout the entire training, the detector is penalized for drawing bboxes accurately with respect to the subject but not in reference to the test set. This is evident, looking at MAP curves in reference to the evaluated CP values, like DPS 9 at MAP 6.35% achieving a CP of 87.1%, Table 5. The algorithm can minimize loss and produce accurate detectors, but it is not unreasonable to think that training may suffer under the described conflict in evaluation. Further evidence supporting this hypothesis is EORS, the only model trained on low-resolution imagery 608x608, achieving high AP and MAP with steep MAP curves, Appendix M and Table 7. It is harder for the detector to adapt differentiating labeling strategies as low-resolution imagery displays fewer objects. With fewer objects, the number of possible interpretations of labeling strategies is reduced with objects being clearly defined, Appendix O.

The IoU evaluation proposes a conflict, but it still provides an essential trait by training predictions to be spatially accurate. Adjusting for IoU ensures that the detector is guided towards remaining spatially accurate, promoting the F1 score and presumably ensuring that emphasis remains on the object, not the bbox backdrop. It would be interesting to alter YOLOv4's IoU threshold while training to determine the effect of lowered IoU and likewise see the result of increasing IoU during training. This way, the detector can both be cut some slack by lowering IoU, and by increasing it, we can try to force it to adapt the labeling parts bbox strategy. These assumptions are simply that and can only be verified through a proper investigation of the impact of IoU during training.

The described conflict highlights the weakness of using YOLOv4 for the given plant species in this study, as defining where to separate subjects as distinct entities can be difficult. An image segmentation algorithm would presumably have been more efficient at localizing subjects by being able to draw regions rather than bboxes. Regions can specify population presence rather than single instances and would be suited better for large beach rose clusters. In contrast, the object detector would probably fair better in dispersed single instances like hogweed. Incorporating the custom coverage measures in evaluation is more in line with that expected for evaluating a segmentation algorithm rather than a detector. Presumably, a segmentation algorithm would produce better results when assessing subjects as coverage rather than single entities, and even for single hogweed instances outperform YOLOv4 in coverage evaluation.

Reflecting on further points of interest

Aspects of color composition were not investigated in the study. Only the augmentation of saturation, brightness and HUE was altered at stable values, which were deemed "not too intrusive", Appendix H. This was a somewhat naive approach as the detector can benefit from color space transformations that a human would deem "uncharacteristically" for the subject at hand, [Gowda and Yuan, 2019]. OpenCV allows for easy color space conversion, and investigating the impact of conversions on YOLOv4 detectors would be relatively straightforward, [Bradski, 2000]. For finished detectors, the imagery would need to be transmuted to the new color space before inference and right back to RGB afterward for proper displaying of imagery. This should have little impact on image inference, but when inferring videos and being time-sensitive towards frame rate, the color space conversion can lead to diminished inference speed. Decreased inference speed can be deemed acceptable if the detector gains ground in its accuracy.

Normalized difference vegetation index (NDVI) is a popular remote sensing index used in assessing vegetation health relying on near-infrared imagery, [Myneni et al., 1995, p. 482]. Using NDVI is not applicable on a standard UAV like the P4 with only an RGB sensor. It is not impossible to get hold of near-infrared sensors, but it is a costly add-on to the platform that must be purchased separately. With NDVI, image values range from -1 to 1, with -1 to 0 being inanimate/dead objects and values above 0 being active photosynthesizing objects. In the range of 0 to 1, intensity roughly equates to the health of plants, with the healthiest topping at the value of 1. Coupled with plants' physical properties and spatial distribution, it can become easier to distinguish between subjects of different classes. This also makes it possible for the detector to disregard the backdrop if it has no photosynthetic activity, like in cases of beach meadow with a rock backdrop, presumably leading to increased detector accuracy.

Concerns about the algorithm overfitting to the background and backdrops like shadows stem from this being a common issue in image classification. This has previously been highlighted during the discussion of performance as to how the nadir view can complicate differentiation of subject and surroundings and how the model's training emphasis on IoU is guided towards focusing on the subject rather than the backdrop. The detector trains on what it is given, and if objects within bboxes are represented often with a specific background or a shadow pointing north-east, then these "static" traits are treated as inherent properties of the object. The key to preventing this is tightening bboxes around labeled subjects or securing variance in the object's backdrop. Sourcing multiple data sets for training would be a reasonable solution. Different data augmentation measures could also be applied to transform pixel values; however, this will also affect the subject itself. A possible but time-consuming effort would be to construct images with randomized backgrounds by cutting subjects from source imagery and laying them atop new backgrounds using a photo editor. This would not be feasible for a whole training set but could be helpful during the validation phase to validate detector integrity in scenarios outside the training area.

Aspects related to biology

In evaluating detector performance, some biological aspects need to be addressed when proclaiming the hogweed and beach rose detectors sufficient for deployment. The vegetative state of plants differs with seasonality. Plant color, size, floral state, and overall health varies greatly throughout seasons, and the vegetative states trained for in detectors only qualify for one window in time out of several. Hogweed and beach roses bear flowers; these are not present in trained sets. As described previously, sea kale has withered flowers in the data set but bears white flowers when blossoming at other times of the year. Ideally, data sets would cover multiple seasonal states to ensure detector robustness in classifying the target no matter the date of flight. "Luckily" the DEPA survey time frame is set from May through October[DEP, 2016b] and the time frame for collecting viable data is "only" half a year. Given the needed knowledge on the plant subject, one can quickly hone in on a survey area to collect data to document all vegetative states throughout the season. Presumably, more robust detectors can be trained if plant life cycles are emphasized and time set aside for a thorough data collection.

In relation to all results, a proper field protocol is required to validate findings - both detections and the detector itself. Lacking field protocols strictly speaking invalidates detectors as no proper documentation is provided for the findings being accurate without groundtruthing. As described for lacking the height of objects during labeling, the same goes for validating findings, as one cannot be sure that a strip of beach roses is not a strip of trees. In ensuring the detector is adequately trained on documented material, further usage of detectors can be conducted without the same level of documentation needed once the detector is validated.

In comparison to related studies

Purposing an object detector for environmental and agricultural services is a field with great potential. To conclude the evaluation of the study and its findings, two closely related studies in palm tree detection will be examined to evaluate the state and viability of the presented detectors. Ammar et al. explore the classification of palm trees across four detectors: Faster R-CNN, Efficient Det-D5, YOLOv3 and YOLOv4[Ammar et al., 2021]. Nair et al. in a publicly available and not peer-reviewed preliminary article, explore how to purpose YOLOv4 for the detection of palm tree canopies[Nair et al., 2021].

Ammar et al. construct a YOLOv4 detector purposed for classifying palm trees and "other" trees. Images are of high resolution at 4864x3648, 4096x2160, and 4000x3000 with YOLOv4's input size set to 608x608, an image- and input size relatable to those used in this study. Images stem from a palm tree plantation survey and a survey above the Prince Sultan University campus (Saudi Arabia). The study at an IoU of 0.5 achieves MAP at 0.96% for palm trees and 0.8% for other trees with precision scores of 0.83 and 0.82 respectively, at 5.43FPS. Detectors are satisfactory in their performance and ready for deployment given their high precision - the frame rate is arguably too slow for video inference. The detector is more efficient in detecting palm trees, and this is most likely due to their distinct star-like shape, whereas other canopies can be easily confused with bushes at nadir view. Focusing on MAP values, the detectors presented by Ammar et al. surpass all detectors within this study. For the presented sample imagery, trees are arranged in rows with spacing in the plantation and on the campus site, making them easily distinguishable from the backdrop. One would assume this to be of benefit during training as objects are clearly defined with no overlap.

Nair et al. explore two data sets: a high-resolution set at 3000x4000 and later a set with images cut to 544x544. The study follows a similar process to what has been presented for EOR and EORS. Nair et al. construct a high-resolution detector with presumed input size 544x544 that ends up fairing unsatisfactory with MAP 47%. Nair et al. point to the labeling of the high-resolution imagery being unable to enclose only the subject within the bboxes and palm trees displaying too varied biological and morphological attributes. Being unsatisfied with the result, images are cut into smaller segments of 544x544 to match the input size of YOLOv4:
"*To ensure the images are compatible with the input for the YOLOv4 pipeline*"[Nair et al., 2021, p 12]. Recall that SPP downsizes imagery to the input size of YOLOv4, and there is no need to purpose imagery to match the input size. This is proven by resolutions incorporated by Ammar et al. and within this study where DPS models at the image resolution 5472x3648 and input size 416x416, Table 1, are concluded to fare with adequate precision.

Nair et al. train a new detector on image cutouts validating with a MAP of 95.65%, directly comparable to EORS with an AP for beach roses at 94.78%. EORS was tested for high-resolution imagery, Table 8, as a detector being able to only infer on 608x608 images is insufficient as this image size is unfeasible for use with UAVs. This would lead to the operator having to bring multiple batteries on surveys for live detection. Otherwise, the detector is only applicable for use with image cutouts in later image analysis at a workstation. The test on variable resolutions showed EORS being unable to infer and detect high-resolution imagery with the resolution 1216x1216, seeing the first drop in F1 score, Table 8, Appendix O. The same issue is observed by Nair et al. at a resolution of 750x1000. However, this does not discourage them from further promoting the use of the detector. Nair et al. proceed with this "best" detector and do not acknowledge the resolution aspect at all and highlight it having video inference at 15.9fps measured by observing the time it takes to infer a single 416x416 image on their given workstation. The article proceeds to compare their detector's accuracy and fps to other high-resolution detectors in literature, proclaiming their proposed detector the superior one[Nair et al., 2021, p 17]. Ammar et al. achieve higher AP with significantly higher image resolution but is not incorporated in Nair et al.'s comparison as both studies have publishing dates within a month of each other.

From a meta-perspective, it is evident that in comparison, Ammar et al. outperform findings in this study and that of Nair et al. when assessing MAP. Ammar et al. achieve high MAP for high-resolution models, whereas similar detectors within this study fail to achieve similar MAP. Why this is the case is not known, but similar MAP should be strived for in purposing detectors for habitat surveillance. In this and Nair et al.'s study, low-resolution detectors are presented with high MAP and, unfortunately, cannot infer and detect objects in high-resolution imagery. Nair et al., however, fail to recognize that usage of the detector is limited to only 416x416 images and videos. The EORS detector was deemed insufficient for this very reason, as the inability to infer high-resolution imagery limits the usage of the detector in inferring video material. Nair et al. also display an inadequate understanding of how YOLOv4 takes input imagery by claiming that the cropping of imagery to 416x416 is needed to fit imagery for YOLOv4's input dimension properly. Nair et al., in contrast to this study, stand as an example of how not remaining critical of one's work can lead to inaccurate or misleading conclusions and proposals. It results in the study standing in a more positive light while undermining the scientific process. By not proposing a new hypothesis when observing lackluster detector performance and relying on observations as groundwork for future detectors, Nair et al. proclaim the results to surpass detectors in related studies display a lack of respect for the scientific method.

Future Prospects

The product as it stands

This study proposes two detectors for deployment: a hogweed detector (DPS2) and a beach rose detector (EOR), achieving coverage precision of 87.9% and 83%, respectively. Detectors yield high precision in detecting coverage of subjects but display different localization capabilities. Hogweed is located accurately with an F1 score of 0.95 and with the beach rose detector lacking localization efficiency at an F1 score of 0.64. Both detectors were trained on clear bbox labels of class subjects no matter their life cycle phase and state. A solid and reliable sea kale detector was not trained sufficiently for deployment. Across the three detector types, collectively 1431 hours of training has been conducted across 44 trained detectors of which 24 are presented in the study, Appendix P. The detectors are presented as trained weights along with a python script, allowing for visualization and calculation of detection coverage relying on the imagery's spatial resolution. In Darknet, imagery and videos can be inferred but not reported on for coverage measures. The aspect of video inference was not investigated further for trained detectors in this study.

The detectors along the presented workflow stand as a proof of concept of YOLOv4 being able to detect custom classes in UAV imagery and videos at nadir view when inferring on a computer workstation. Detectors are plural, and thus two separate weight files were trained for single classes. Ideally, the detector should be trained collectively on all classes to make a single detector for usage rather than separate detectors depending on the habitat. Proper application of a YOLOv4 detector should strive for the detector to be multi-class. A single multi-class detector promotes automation of findings, reduces computation time, and does not make the process unnecessarily complex for users. Being prompted only to use a single multi-class detector eliminates the need for biological and ecological domain knowledge in utilizing the detector workflow.

In relation to DEPA efforts

In briefly recapping the process of terrestrial habitat surveillance under the DEPA, *in situ* surveys of documentation sites are conducted in circles of five meters in diameter by expert personnel. Delegates documenting within select large habitats chosen for detailed survey annotate all plants found within documentation sites and additionally any rare or invasive species encountered outside the site by coincidence. Relying on *in situ* annotations, satellite imagery of Danish habitats is visually inspected in a GIS. Habitats are annotated with polygons and classified depending on what nearby documentation sites have noted on species composition and from what little can be seen using low spatial resolution satellite imagery, Figure 1a&b.

Comparatively, the presented detectors rely on high-resolution imagery at a high spatial resolution, Figure 1c, providing high-resolution image documentation of UAV surveys. The UAV surveys far exceed the reach of documentation sites. Assuming battery life of 22 minutes, classifying hogweed at an altitude of 8m at medium flight speed with 70% image overlap in a square will result in a documentation survey of 8,000m^2 compared to current circular documentation sites spanning 20m^2. With the same settings, the sea kale detector at 20m altitude will span 28,000m^2 and the beach rose detector at 100m altitude span a survey area of 555,000m^2. The spatial reach of the UAV and the provision of detailed image documentation is undeniably far more valuable than the documentation and coverage produced by *in situ* documentation site inspections.

In favor of manual classification of species within documentation sites, the experts can classify far more plant species than highlighted detectors. Suppose ambiguous samples of species are found to be in a state challenging to classify. In that case, the experts can bring samples back from the field or further investigate the subject, whereas the detector would simply overlook the subject. Proposed detectors are a proof of concept that YOLOv4 can become a reliable detector in habitat surveillance. Ideally, detectors will be trained for all species used by DEPA to classify habitats[DEP, 2016a]. Given the capability to classify as many species and perhaps detect more within a given area renders the UAV detectors the triumphant surveillance method as the capturing of photographic documentation and spatial reach is superior to manual labor. Another benefit of using the detector is reducing the variance induced by different DEPA-appointed delegates performing surveys and having different skills and biases associated with their assessments. With the detector being the sole "artificial delegate" performing surveys, a precise measure is provided for inaccuracy and performance to rely on in reviewing findings. Having the detector being the delegate results in the UAV operator not needing any skill sets that the usual DEPA delegate possesses. The detector is responsible for doing the delegate's work, and it only relies on a UAV operator to conduct the flight.

Alternatively, if we chose not to trust detectors fully, they could prove capable as inspection tools. Inferring on video and detecting live in the field can help indicate where documentation sites should be established. Additionally, the detector can report on any rare or invasive species for the operators to additionally note during their survey while the UAV, during its guided search, looks for a documentation site. This would require real-time inference on video material, an aspect not investigated in this study. Another aspect not investigated in this study which is a requirement for DEPA to digitize results efficiently is the implementation of GIS digitizing of findings. Storing mass amounts of pictures of detections is not sustainable and would require too many resources for what little information the images themselves convey. Calculated coverage and biomass of detections convey useful information, but they convey little information without being registered with respect to their spatial distribution. The demand for a GIS storage solution for results allows for spatial representation and calculation of detections and remains an essential requisite to report findings for EU assessment. In the current workflow, DEPA manually digitizes habitat nature, and ideally, this should become automated to free up resources for use in other important tasks.

GIS implementation

The workflow for constructed detectors lacks digitization of findings in GIS. This is required to efficiently store findings as shapefiles depicting detections as polygons rather than storing raster images with depicted detections. Images can be geotagged and presented in a GIS in the location they were taken. Using the detector and drawing detections in shapefiles would efficiently digitize detections in a georeferenced shapefile. Such is the case with the Deep Learning Library[Redlands, 2019] for ArcGIS [Redlands, 2010], a paid license GIS software suite. ArcGIS allows for the training of YOLOv3 detectors and other segmentation and detection algorithms within the GIS interface, promoting the user experience while lowering entry barriers for users' skill sets. Labeling, training, and validating take place within ArcGIS, and deploying the subsequent detector is as easy as clicking a button.

Drawing inspiration from ArcGIS, a similar implementation can be proposed in freeware using Qgis. Qgis is an open-source GIS where developers can write and share custom tools written in either python or C++, [QGIS Development Team, 2022]. The python script for coverage assessment mounts the detector using OpenCV and draws detections using shapely. Shapely geometry can be saved as shapefiles. A proposition for a freeware-based tool in Qgis would be to adopt a workflow as depicted in Figure 6. Imagery should be imported and stitched together into an orthomosaic. Such a process/tool is currently unavailable in Qgis, but propositions for such tools are explored within the plugin community[ODM Community, 2020]. The orthomosaic is georeferenced and oriented to match the map projection for which the GIS is viewed. Using OpenCV, the select detector is mounted. For the appropriate detector image size, i.e., the one it was trained for, the orthomosaic is split into cutouts at specific anchor points. Ideally, these cutouts are temporarily stored as NumPy arrays based on set anchor points on the raster orthomosaic with the detector inferring in the single cutout. When done, a new temporary cutout is grabbed and inferred upon. Simultaneously detections for each cutout are drawn in a separate shapefile layer using Shapely to store detections in the GIS. Shapefiles can further be supplied during the inference process with class id, confidence, coverage, and what else may be of interest to log.

Such a workflow will ensure the concept proposed in the study remains freeware-based and allow users to efficiently utilize detectors if the workflow is programmed correctly, Figure 6. This process would work for not only YOLOv4 models but all segmentation algorithms, detectors, and classifiers mountable using the OpenCV library and essentially mirror the paid process distributed for the ArcGIS platform.

Live feed video inspections

For live video inspections using detectors, the P4 and similar platforms can be equipped with onboard computers to handle the feed of video and detector calculations. These onboard computers are small and brand a sufficient central- and graphical processing units. The Raspberry Pi is an example of a small computer

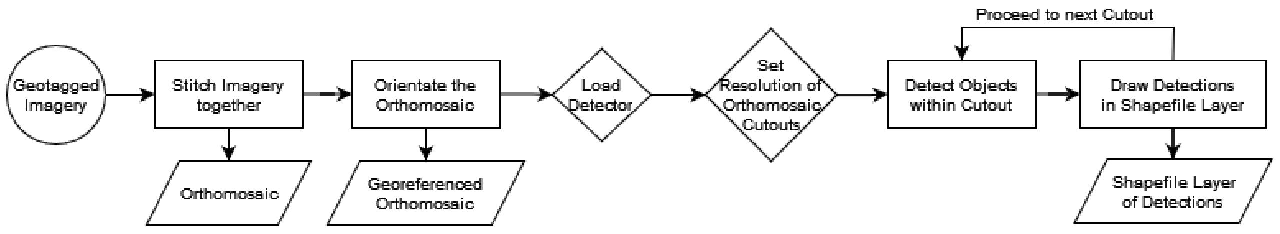
Proceed to next Cutout
Geotagged Imagery
Stitch Imagery together
Orthomosaic
Orientate the Orthomosaic
Georeferenced Orthomosaic
Load Detector
Set Resolution of Orthomosaic Cutouts
Detect Objects within Cutout
Draw Detections in Shapefile Layer
Shapefile Layer of Detections

Figure 6: *Proposed workflow for digitizing YOLOv4 detections in a python based Geographical Information System. Circles symbolize the start point, diamonds decisions, squares steps, and last trapezes symbolize data outputs.*

that is both cheap and capable with a low weight[Raspberry-Pi, 2022]. In higher weight classes, Nvidia's Jetson computers are small but very powerful with high-grade graphics processors[NVIDIA, 2022b]. For UAVs like P4, the loading weight of the platform is the limiting factor in choosing what computer to use; the heavier the computer, the larger the UAV. P4 can only carry approximately 450g safely, and having a small frame size is limited in the use of onboard computers.

Ayoub and Schneider-Kamp explore the use of different onboard computers to deploy YOLO in detecting power line components[Ayoub and Schneider-Kamp, 2021]. Using TensorRT to optimize trained weights for video inference, YOLO detectors are displayed efficiently inferring at adequate frame rates up to 50fps. These high rates are achieved for the YOLO tiny models and on the mid-range powerful Nvidia mini-computer AGX Xavier[Ayoub and Schneider-Kamp, 2021]. Results are thus not directly comparable to anything within this study but prove the possibility for easy application of detectors in video material during live inspections. Ideally, the detectors for this study should be investigated using YOLO tiny models to achieve similar frame rates. This study proves that it is possible to accurately detect plant species in habitat surveys; integrating these into live video feed inference is the next central step.

Based on these findings and relying on Ayoub and Schneider-Kamp's work[Ayoub and Schneider-Kamp, 2021] it should not prove too difficult to investigate and propose a complete workflow for an operational video platform. This would allow DEPA and other interests to detect objects in real-time and use the detectors as a tool *in situ* to promote their efforts. Being made aware of the presence of rare or invasive species across large habitats allows resources to be used efficiently to focus efforts on these points of interest. This can further promote work towards habitat conservation plans and efforts in fighting invasive species. These opportunities yield great value and impact that might otherwise have been overlooked *in situ* when surveying manually within a single documentation site.

Alternative algorithms

The choice of using YOLOv4 in this study was mainly based on YOLOv4's speed of inference that allows for analysis of video material in real time[Bochkovskiy et al., 2020, p. 10]. If time permitted, it would be interesting to test other object detectors and segmentation algorithms for what performance they would have in comparison to presented YOLOv4 detectors. The choice between algorithms is often a cost-benefit situation between increasing precision at the cost of inference speeds. If one desires to infer on video, then high inference speeds are a must at the cost of detection accuracy. YOLOv4, as highlighted by its authors [Bochkovskiy et al., 2020, p. 11-13], is less accurate than EfficientDet [Tan et al., 2019], SM-NAS [Yao et al., 2020], ATSS [Zhang et al., 2020], NAS-FPN [Ghiasi et al., 2019], CenterMask [Lee and Park, 2019], TridentNet [Li et al., 2019], Centernet [Duan et al., 2019], RPDet [Yang et al., 2019] and SAPD [Zhu et al., 2019] when only assessing accuracy and not inference speeds. Even as YOLOv4 loses ground in raw accuracy, the detector remains renowned for its "high accuracy - high inference" reputation, and surpassing it is difficult. A rivaling candidate to YOLOv4 when considering the need for video inference would be YOLACT, You Only Look At CoefficienTs [Bolya et al., 2019].

YOLACT is a hybrid between a segmentation algorithm and object detector; bboxes are drawn around objects that additionally themselves are covered by a drawn region. It infers at high frame rates like YOLOv4 and drawing detections as regions find it being better purposed for digitizing detections in GIS. Rather than digitizing large bboxes around large objects resulting in gross overestimations, Appendix N, regions are more true to what the detector has found and what should be depicted in the GIS, Appendix Q. If one wishes to increase accuracy and GIS digitization quality at the cost of losing video inference, then Mask R-CNN should be considered a top candidate. Mask R-CNN is a segmentation algorithm yielding high accuracy measures at a severe cost of inference rates, leaving it unsuitable for video inference but still capable of computing still imagery at satisfactory rates[He et al., 2017].

Alternatives to YOLOv4 are aplenty, and it should be valued what the user desires in their image classification task. It is a costs and benefits valuation with object detectors like YOLOv4 and YOLACT winning when having a need for video inference and Mask R-CNN triumphing if accuracy is in demand, no matter the cost of video inference. The value gained through video inspections in detecting points of interest that otherwise would be overlooked is deemed an important factor in the choice of using an object detector. If DEPA was not tasked with planning conservation care plans for habitat nature, then losing detections of sensitive species' sparse presence would not be as detrimental. This would instead promote the use of a segmentation algorithm as Mask R-CNN as the emphasis would be on reporting on the temporal status of habitats and their spatial reach. With DEPA's mission being ambitious, this is not the case. The localization features that object detectors provide are of great value to secure accurate habitat surveillance reports under The European Union's Habitat Directive.

Conclusion

YOLOv4 was successfully trained to detect sea kale (*Crambe maritima*), hogweed (*Heracleum mantegazzianum*), and beach roses (*Rosa rugosa*) in UAV imagery at nadir view relying solely on open-source solutions. Mean average precision (MAP) values of detectors upon concluding training indicate all being inadequate at MAP $< 45\%$. Incorporating custom measures to evaluate coverage of detections in relation to validation material sees detectors being precise in detecting and accurately estimating subject coverage. Hogweed was detectable at a coverage precision of 87.9%, sea kale at 52.8%, and beach roses at 83% at altitudes of 8m, 20m, and 100m, respectively. Detectors for hogweed and beach rose are deemed precise and adequate for deployment. Sea kale falls short at a coverage precision below 80% and is deemed inadequate in serving as a reliable detector. For the viable detectors, single entities of hogweed can be efficiently localized with an F1 score of 0.9, whereas single beach roses entities are difficult to localize, scoring 0.64 in its F1 score.

All detections and evaluations are based on detectors being compiled on a workstation computer, not a UAV-associated small factor computer platform. An implementation for digitizing detection in a geographical information system (GIS) is proposed along a workflow for deploying detectors in the field relying on small factor computers for video processing on a UAV platform. This study proves that YOLOv4 is a capable tool in habitat surveillance efforts, as documented by the two deployment-ready detectors presented. Relying on UAVs for data input, detectors allow for detailed and accurate documentation of vegetation detection and classification. Unlike relying on *in situ* inspections and manual assessment of satellite imagery, the UAVs provide a high-resolution data foundation to document and digitize key plant species presence within Danish habitat nature. The proposed use of detectors as high-resolution image detectors, real-time video inspection platforms, and GIS digitizers is presented as a solution for the European Union to succeed in upholding The European Habitat Directive internationally and globally to ensure that The Sustainable Development Goals of the United Nations are achieved.

References

P. Adarsh, P. Rathi, and M. Kumar. Yolo v3-tiny: Object detection and recognition using one stage improved model. pages 687–694, 03 2020. doi: 10.1109/ICACCS48705.2020.9074315.

A. Al-Kaff, D. Martin, F. Garcia, A. de la Escalera, and J. Maria Armingol. Survey of computer vision algorithms and applications for unmanned aerial vehicles. *EXPERT SYSTEMS WITH APPLICATIONS*, 92:447–463, FEB 2018. ISSN 0957-4174. doi: {10.1016/j.eswa.2017.09.033}.

M. A. Al-masni, M. A. Al-antari, J.-M. Park, G. Gi, T.-Y. Kim, P. Rivera, E. Valarezo, M.-T. Choi, S.-M. Han, and T.-S. Kim. Simultaneous detection and classification of breast masses in digital mammograms via a deep learning YOLO-based CAD system. *COMPUTER METHODS AND PROGRAMS IN BIOMEDICINE*, 157:85–94, APR 2018. ISSN 0169-2607. doi: {10.1016/j.cmpb.2018.01.017}.

A. Ammar, A. Koubaa, and B. Benjdira. Deep-learning-based automated palm tree counting and geolocation in large farms from aerial geotagged images. *Agronomy*, 11:1458, 07 2021. doi: 10.3390/agronomy11081458.

N. Ayoub and P. Schneider-Kamp. Real-time on-board deep learning fault detection for autonomous uav inspections. *Electronics*, 10(9), 2021. ISSN 2079-9292. doi: 10.3390/electronics10091091. URL https://www.mdpi.com/2079-9292/10/9/1091.

J. A. J. Berni, P. J. Zarco-Tejada, L. Suarez, and E. Fereres. Thermal and Narrowband Multispectral Remote Sensing for Vegetation Monitoring From an Unmanned Aerial Vehicle. *IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING*, 47(3, SI):722–738, MAR 2009. ISSN 0196-2892. doi: {10.1109/TGRS.2008.2010457}.

T. Blaschke. Object based image analysis for remote sensing. *ISPRS JOURNAL OF PHOTOGRAMMETRY AND REMOTE SENSING*, 65(1):2–16, JAN 2010. ISSN 0924-2716. doi: {10.1016/j.isprsjprs.2009.06.004}.

A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao. Yolov4: Optimal speed and accuracy of object detection, 2020. URL https://arxiv.org/abs/2004.10934. Source code; https://github.com/AlexeyAB/darknet.

D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee. YOLACT++: better real-time instance segmentation. *CoRR*, abs/1912.06218, 2019. URL http://arxiv.org/abs/1912.06218.

G. Bradski. The opencv library. *Dr. Dobb's Journal of Software Tools*, 2000.

M. Bryson, M. Johnson-Roberson, R. J. Murphy, and D. Bongiorno. Kite Aerial Photography for Low-Cost, Ultra-high Spatial Resolution Multi-Spectral Mapping of Intertidal Landscapes. *PLOS ONE*, 8(9), SEP 19 2013. ISSN 1932-6203. doi: {10.1371/journal.pone.0073550}.

P. E. Carbonneau and J. T. Dietrich. Cost-effective non-metric photogrammetry from consumer-grade suas: implications for direct georeferencing of structure from motion photogrammetry. *Earth Surface Processes and Landforms*, 42(3):473–486, 2017. doi: https://doi.org/10.1002/esp.4012. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/esp.4012.

X. Chen, L. Qi, Y. Yang, Q. Luo, O. Postolache, J. Tang, and H. Wu. Video-Based Detection Infrastructure Enhancement for Automated Ship Recognition and Behavior Analysis. *JOURNAL OF ADVANCED TRANSPORTATION*, 2020, JAN 20 2020. ISSN 0197-6729. doi: {10.1155/2020/7194342}.

S. Connor and K. Taghi. A survey on image data augmentation for deep learning. *J. Big Data*, 6:60, 2019. doi: 10.1186/s40537-019-0197-0. URL https://doi.org/10.1186/s40537-019-0197-0.

H. P. Damberg. Phantom 4 pro, 2022. URL https://www.dji.com/dk/phantom-4-pro/info. Accessed: 2022-05-17.

N. C. f. M. o. E. DCE. Bevaringsstatus for naturtyper og arter, oversigt over danmarks artikel 17-rapportering til habitatdirektivet 2019. Note on EEA findings for distribution to interests, 2019.

Key for identification of danish nature types under the habitatdirective - Nøgle til identifikation af danske naturtyper på habitatdirektivet. DEPA, Danish Environmental Protection Agency, Tolderlundsvej 5, 5000 Odense, MAY 2016a.

Mapping of terrestrical, full sun nature types - Kortlægning af terrestriske, lysåbne habitatnaturtyper. DEPA, Danish Environmental Protection Agency, Tolderlundsvej 5, 5000 Odense, MAR 2016b.

Habitat descriptions, 2016 - Habitatbeskrivelser, årgang 2016. DEPA, Danish Environmental Protection Agency, Tolderlundsvej 5, 5000 Odense, MAY 2016c.

DJI. Phantom 4 pro, 2016. URL https://www.dji.com/dk/phantom-4-pro/info. Accessed: 2022-02-16.

DJI. Phantom drone series, 2022. URL https://www.dji.com/dk/products/phantom. Accessed: 2022-02-23.

K. du Preez. Yolov4 community discussion purposing using uav imagery, 2019. URL https://github.com/pjreddie/darknet/issues/1535. Forum post with no findings supplied to backup the claim. An online thread that I have no association with. Accessed: 2022-04-20.

K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

E. E. A. EEA. *State of nature in the EU, Results from reporting under the nature directives 2013-2018*. Kongens Nytorv 6, Copenhagen, 2020.

EU. Council Directive 92/43/EEC of 21 May 1992 on the conservation of natural habitats and of wild fauna and flora OJ L 206, 22.7.1992. pages 7–50, JUN 1992.

EU. Interpretation Manual of European Union Habitats, version EUR 28. pages 7–142, JAN 2013.

K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36:193–202, 02 1980. doi: 10.1007/BF00344251.

G. Ghiasi, T. Lin, R. Pang, and Q. V. Le. NAS-FPN: learning scalable feature pyramid architecture for object detection. *CoRR*, abs/1904.07392, 2019. URL http://arxiv.org/abs/1904.07392.

S. N. Gowda and C. Yuan. Colornet: Investigating the importance of color spaces for image classification. *CoRR*, abs/1902.00267, 2019. URL http://arxiv.org/abs/1902.00267.

K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *Computer Vision – ECCV 2014*, pages 346–361. Springer International Publishing, 2014. doi: 10.1007/978-3-319-10578-9_23. URL https://doi.org/10.1007%2F978-3-319-10578-9_23.

K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017. URL http://arxiv.org/abs/1703.06870.

C. Holness, T. Matthews, K. Satchell, and E. C. Swindell. Remote Sensing Archeological Sites through Unmanned Aerial Vehicle (UAV) Imaging. In *2016 IEEE INTERNATIONAL GEOSCIENCE AND REMOTE SENSING SYMPOSIUM (IGARSS)*, IEEE International Symposium on Geoscience and Remote Sensing IGARSS, pages 6695–6698. Inst Elect & Elect Engineers; Inst Elect & Elect Engineers, Geoscience & Remote Sensing Soc; NSSC, 2016. ISBN 978-1-5090-3332-4. doi: {10.1109/IGARSS.2016.7730748}. 36th IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, PEOPLES R CHINA, JUL 10-15, 2016.

G. Jocher, A. Stoken, A. Chaurasia, J. Borovec, NanoCode012, TaoXie, Y. Kwon, K. Michael, L. Changyu, J. Fang, A. V, Laughing, tkianai, yxNONG, P. Skalski, A. Hogan, J. Nadar, imyhxy, L. Mammana, AlexWang1900, C. Fati, D. Montes, J. Hajek, L. Diaconu, M. T. Minh, Marc, albinxavi, fatih, oleg, and wanghaoyang0106. ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support, Oct. 2021. URL https://doi.org/10.5281/zenodo.5563715.

Y. Jung, H. Bang, and D. Lee. Robust marker tracking algorithm for precise uav vision-based autonomous landing. In *2015 15th International Conference on Control, Automation and Systems (ICCAS)*, pages 443–446, 2015. doi: 10.1109/ICCAS.2015.7364957.

K. Koech. Confusion matrix for object detection, 2020. URL https://towardsdatascience.com/confusion-matrix-and-object-detection-f0cbcb634157. Accessed: 2022-05-30.

A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012. doi: 10.1145/3065386.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278 – 2324, 12 1998. doi: 10.1109/5.726791.

Y. Lee and J. Park. Centermask : Real-time anchor-free instance segmentation. *CoRR*, abs/1911.06667, 2019. URL http://arxiv.org/abs/1911.06667.

Y. Li, Y. Chen, N. Wang, and Z. Zhang. Scale-aware trident networks for object detection. *CoRR*, abs/1901.01892, 2019. URL http://arxiv.org/abs/1901.01892.

S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia. Path aggregation network for instance segmentation, 2018. URL https://arxiv.org/abs/1803.01534.

M. Loey, G. Manogaran, M. H. N. Taha, and N. E. M. Khalifa. Fighting against COVID-19: A novel deep learning model based on YOLO-v2 with ResNet-50 for medical face mask detection. *SUSTAINABLE CITIES AND SOCIETY*, 65, FEB 2021. ISSN 2210-6707. doi: {10.1016/j.scs.2020.102600}.

A. Menon, B. Omman, and S. Asha. Pedestrian Counting Using Yolo V3. In *2021 INTERNATIONAL CONFERENCE ON INNOVATIVE TRENDS IN INFORMATION TECHNOLOGY*

(ICITIIT), 2021. ISBN 978-1-6654-0467-9. doi: {10.1109/ICITIIT51526.2021.9399607}. International Conference on Innovative Trends in Information Technology (ICITIIT), Kottayam, INDIA, FEB 11-12, 2021.

D. Merkel. Docker: lightweight linux containers for consistent development and deployment. *Linux journal*, 2014(239):2, 2014.

D. Miljøportal. Denmarks area information, 2022. URL https://arealinformation.miljoeportal.dk/html5/index.html?viewer=distribution. Accessed: 2022-02-22.

R. Mlambo, I. H. Woodhouse, F. Gerard, and K. Anderson. Structure from motion (sfm) photogrammetry with drone data: A low cost method for monitoring greenhouse gas emissions from forests in developing countries. *Forests*, 8(3), 2017. ISSN 1999-4907. doi: 10.3390/f8030068. URL https://www.mdpi.com/1999-4907/8/3/68.

N. Mohamed, J. Al-Jaroodi, I. Jawhar, A. Idries, and F. Mohammed. Unmanned aerial vehicles applications in future smart cities. *TECHNOLOGICAL FORECASTING AND SOCIAL CHANGE*, 153, APR 2020. ISSN 0040-1625. doi: {10.1016/j.techfore.2018.05.004}.

R. Myneni, F. Hall, P. Sellers, and A. Marshak. The interpretation of spectral vegetation indexes. ieee trans. geosci. remote sens. *Geoscience and Remote Sensing, IEEE Transactions on*, 33:481 – 486, 04 1995. doi: 10.1109/36.377948.

K. Nair, S. Chakkaravarthy, R. Dhulipalla, S. Satapathy, and A. Kanungo. Modified yolov4 for real-time coconut trees detection from an unmanned aerial vehicle. *Agronomy*, PREPRINT: 19, 08 2021. doi: 10.21203/rs.3.rs-826223/v1.

NVIDIA. Docker image; nvidia/cuda:11.3.0-cudnn8-devel-ubuntu20.04. https://hub.docker.com/layers/cuda/nvidia/cuda/11.3.0-cudnn8-devel-ubuntu20.04/images/sha256-a39884f8b39cb50f335881ef8662a3f21aa1be6a5087842f6baec81891f20501?context=explore, 2022a.

NVIDIA. Advanced ai embedded systems, 2022b. URL https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/. Accessed: 2022-05-30.

ODM Community. Opendronemap, 2020. URL https://github.com/OpenDroneMap/ODM.

QGIS Development Team. Qgis geographic information system, 2022. URL https://www.qgis.org.

Raspberry-Pi. Raspbeyy pi products, 2022. URL https://www.raspberrypi.com/products/. Accessed: 2022-05-30.

C. E. S. R. I. Redlands. Arcgis desktop: Release 10, 2010.

C. E. S. R. I. Redlands. Arcgis image analyst extension, 2019.

J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018. URL http://arxiv.org/abs/1804.02767.

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection, 2016.

M. Scharling and T. Abildgaard. February gave water in abundance, 2022. URL https://www.dji.com/dk/phantom-4-pro/info. Accessed: 2022-02-17.

N. Svane, M. R. Flindt, R. N. Petersen, and S. Egemose. Physical stream quality measured by drones and image analysis versus the traditional manual method. *Environmental Technology*, 43(8):1237–1247, 2020. doi: 10.1080/09593330.2020.1824022. URL https://doi.org/10.1080/09593330.2020.1824022. PMID: 32921267.

N. Svane, T. Lange, S. Egemose, O. Dalby, A. Thomasberger, and M. R. Flindt. Unoccupied aerial vehicle-assisted monitoring of benthic vegetation in the coastal zone enhances the quality of ecological data. *PROGRESS IN PHYSICAL GEOGRAPHY-EARTH AND ENVIRONMENT*, 2021. ISSN 0309-1333. doi: {10.1177/03091333211052005}.

M. Tan, R. Pang, and Q. V. Le. Efficientdet: Scalable and efficient object detection. *CoRR*, abs/1911.09070, 2019. URL http://arxiv.org/abs/1911.09070.

Y. Tian, G. Yang, Z. Wang, H. Wang, E. Li, and Z. Liang. Apple detection during different growth stages in orchards using the improved YOLO-V3 model. *COMPUTERS AND ELECTRONICS IN AGRICULTURE*, 157:417–426, FEB 2019. ISSN 0168-1699. doi: {10.1016/j.compag.2019.01.012}.

Trafikstyrelsen. Drone flight in the open category, 2022. URL https://droneregler.dk/%C3%85bne-kategori-og-overgangsregler/Droneflyvning-i-den-%C3%A5bne-kategori#. Danish Civil Aviation and Railway Authority, Accessed: 2022-05-17.

Tzutalin. Labelimg, 2015. URL https://github.com/tzutalin/labelImg. Accessed: 2022-03-04.

UN. The 17 sustainable development goals, 2022. URL https://sdgs.un.org/goals. Accessed: 2022-02-23.

A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis. Deep Learning for Computer Vision: A Brief Review. *COMPUTATIONAL INTELLIGENCE AND NEUROSCIENCE*, 2018, 2018. ISSN 1687-5265. doi: {10.1155/2018/7068349}.

C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, Y.-H. Wu, P.-Y. Chen, and J.-W. Hsieh. Cspnet: A new backbone that can enhance learning capability of cnn, 2019. URL https://arxiv.org/abs/1911.11929.

L. Yang, J. Qi, J. Xiao, and X. Yong. A literature review of uav 3d path planning. In *Proceeding of the 11th World Congress on Intelligent Control and Automation*, pages 2376–2381, 2014. doi: 10.1109/WCICA.2014.7053093.

Z. Yang, S. Liu, H. Hu, L. Wang, and S. Lin. Reppoints: Point set representation for object detection. *CoRR*, abs/1904.11490, 2019. URL http://arxiv.org/abs/1904.11490.

L. Yao, H. Xu, W. Zhang, X. Liang, and Z. Li. Sm-nas: Structural-to-modular neural architecture search for object detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34 (07):12661–12668, Apr. 2020. doi: 10.1609/aaai.v34i07.6958. URL https://ojs.aaai.org/index.php/AAAI/article/view/6958.

S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

X. Zhao, Q. Fei, and Q. Geng. Vision based ground target tracking for rotor uav. In *2013 10th IEEE International Conference on Control and Automation (ICCA)*, pages 1907–1911, 2013. doi: 10.1109/ICCA.2013.6565085.

C. Zhu, F. Chen, Z. Shen, and M. Savvides. Soft anchor-point object detection. *CoRR*, abs/1911.12448, 2019. URL http://arxiv.org/abs/1911.12448.