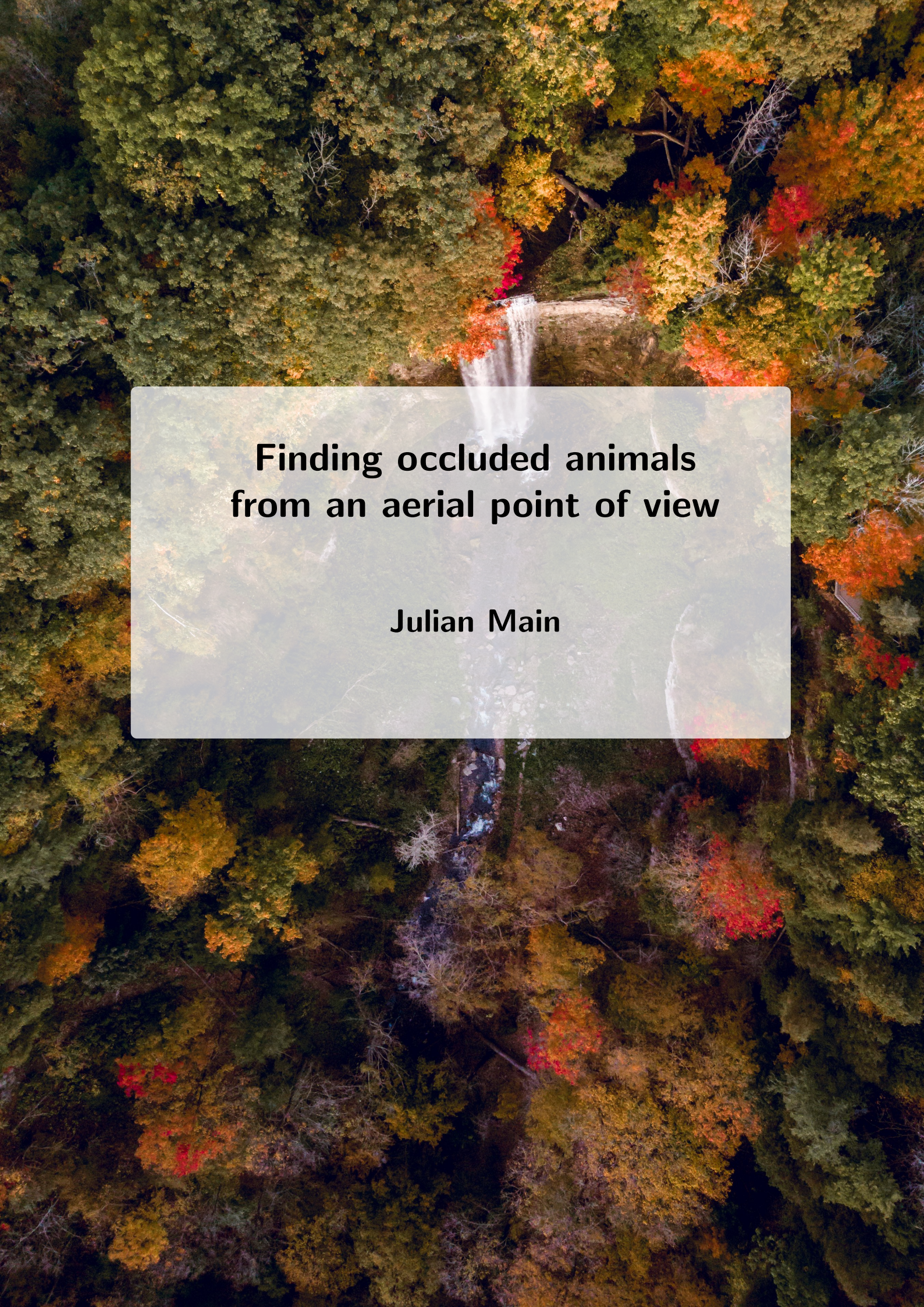


Finding occluded animals from an aerial point of view

Julian Main

Finding occluded animals from an aerial point of view

Julian Main
10541578

Bachelor thesis
Credits: 18 EC

Bachelor Opleiding Kunstmatige Intelligentie

University of Amsterdam
Faculty of Science
Science Park 904
1098 XH Amsterdam

Supervisors
dr. A. (Arnoud) Visser
MSc. Lieuwe Rekker

Informatics Institute
Faculty of Science
University of Amsterdam
Science Park 904
1098 XH Amsterdam

January 31st, 2019

Contents

1 Introduction

This thesis was inspired by the challenge, *Find a lost dog with a drone* [1], which was posed by the European Union research robotics project SciRoc. SciRoc posed multiple challenges with the aim of integrating robots into smart cities. The challenges pushed for robots to cooperate with smart city infrastructure by accomplishing tasks such as assisting people and providing emergency services.

The challenge *Find a lost dog with a drone* describes a scenario in which an aerial robot is given the task of finding a lost dog (which can be represented as a stuffed animal instead of a real dog) inside a mall. The aerial robot must be able to autonomously fly around, while avoiding obstacles, and report the dog's location. The main objective of this challenge is to perform object perception in a cluttered environment.

Aside from finding lost dogs, the use of aerial images to detect objects has many other real world applications, including the tracking and monitoring of animal populations for wildlife protection [Linchant et al., 2015]. These aerial images can be obtained from small, unmanned drones. Unmanned drones can be cheaper, faster, and more accurate than relying on manual, human labor.

Using aerial images can be challenging [Hardin and Jensen, 2011]. One of these challenges is that the objects which we wish to find can move around quickly, which thus requires an object detection algorithm which is capable of working in real-time. However, object detection, especially real-time object detection, requires much computing power which is often not available on current small drones. Thus, to perform real-time object detection, one needs to utilize a lightweight algorithm that can process a high volume of frames per second. Another challenge is acquiring images from an aerial point of view. Since the majority of pictures are not taken from above, it can be hard to acquire training data for training a classifier.

Aside from the object detection part, the control of a drone while keeping an object of interest in the field of vision, is a challenge itself [Rodríguez-Canosa et al., 2012]. Since the drone must be able to move around, avoid obstacles, and simultaneously keep the object of interest in the field of vision. Therefore, this thesis primarily focuses on object detection and purposefully does not discuss the complexities of drone control.

This thesis' principle objective is to analyze the varying methodologies required for finding occluded animals from an aerial point of view. Therefore, the main question which we will achieve to answer is, which methods are suitable for tracking animals from an aerial view? And what kind of (training) data is required for this task.

There exist many different object detection methods [Grauman and Leibe, 2011]. The following sections will briefly discuss some of these methods, after which the experiments which were conducted to test these methods on the task of detecting animals under occlusion, will be given in detail. In conclusion we will discuss the results of the experiments, and which object detection methods are suitable for the task of finding occluded animals.

[1] `https://sciroc.eu/e11-find-a-lost-dog-with-a-drone/`

2 Object detection

The task of detecting objects in digital images has many different applications, including face detection [Viola and Jones, 2004], pedestrian detection [Dollar et al., 2012], and ball detection in soccer matches [Yu et al., 2003]. This thesis will specifically look at detecting occluded animals from an aerial point of view. To achieve this task, we have looked at several object detection techniques. Below we give a short overview of these methods.

2.1 Viola Jones object detection framework

The Viola-Jones object detection framework [Viola and Jones, 2001] was one of the first object detection methods capable of detecting objects in images at multiple frames per second.

Viola-Jones made three contributions. Firstly, a new image representation which was named "Integral Image" was introduced. Integral Image allowed features of objects to be calculated very fast. The Integral Image was defined as an intermediate representation of the image, and was generated by computing the *rectangle features* of the image. Three types of rectangle features were used: a *two-rectangle feature*, a *three-rectangle feature*, and a *four-rectangle feature*. These rectangle features were computed by taking the difference of the sum of pixels within two rectangular regions, the sum within two outside rectangles subtracted from the sum of a rectangle in the center, and the diagonal pairs of rectangles. Compared to other techniques at the time, the rectangles features could be computed relatively fast, and provided a rich representation of features of the objects in the image.

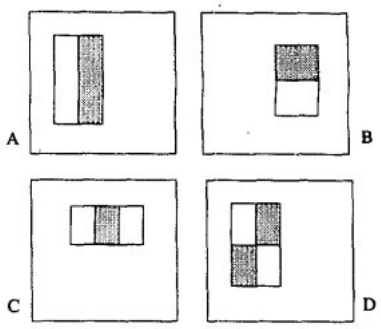


Figure 1: Example of the rectangle features in the Viola-Jones framework. Courtesy of [Viola and Jones, 2001].

The second contribution was a learning algorithm capable of classifying image features. The learning algorithm, based on AdaBoost [Freund and Schapire, 1997], was capable of selecting critical features from a large dataset.

The third contribution was a method which combined complex classifiers. These classifiers allowed certain background regions of the imaged to be removed, so that the focus of the classifier was placed more on regions which were similar to objects.

The framework was tested on images of faces, and was able to perform object detection at 15 frames per second, which was 15 times faster than the fastest algorithm at the time [Rowley et al., 1998].

2.2 Scale-invariant feature transform

Scale-invariant feature transform [Lowe, 2004] is a method for extracting features from images. The extracted features are robust enough to recognize objects in various situations. The objects can be recognized when they are rotated, scaled, changed in illumination, and when masked noise with added.

SIFT can extract larges amount of distinctive features from images. Moreover, it can use an efficient algorithm to reduce the amount of critical features. The algorithm is efficient because it can perform expensive operations only on certain locations which pass initial tests. There are four important steps in SIFT:

1. **Scale-space extrema detection**
   In this stage, a difference-of-Gaussian function is used to search all scales and locations in the image for points which are invariant to scale and orientation.

2. **Keypoint localization**
   At all the locations which were found, a model is fit to find the location and scale. Then keypoints are chosen. The keypoints are chosen on certain measures of stability.

3. **Orientation assignment**
   Orientations are assigned to all the keypoints. With these orientations, an invariance to scale, and location and orientation is found.

4. **Keypoint descriptor**
   In the final step a descriptor is created for each keypoint. These descriptors are capable of matching objects among occlusion, change in illumination, and clutter.

2.3 Histogram of oriented gradients (HOG)

Similar to SIFT, the histogram of oriented gradients (HOG) [Dalal and Triggs, 2005] is a feature descriptor. HOG was originally developed for people detection.

It works by creating a dense grid of uniformly spaced cells, then it extracts gradient based descriptors from windows of interest. These are then classified as belonging to a class, or not belonging to a class ("persons", or non-persons for example). The classification is done by a linear SVM.

The windows of interest on which the classification is done, consist of concatenated gradient histograms. The histograms are formed by dividing subwindows into cells, each of which create a gradient value based on pixel directions. HOG uses 8 x 8 pixel cells and 9 orientations. Using it in this way provides robustness against shifts or rotations. Robustness against illumination is done by normalizing gradient responses before putting them in histograms.

Training is done using a linear SVM, in which the training data consists of positive and negative examples of which the HOG descriptors are extracted.

2.4 Bag-of-words image classification

Bag-of-words [Csurka et al., 2004] is an image classification method which uses words as image features. The words describe local features in images. The training of BoW is done in these steps:

1. Gather training image examples
2. Detect local features
3. At each sampled point, extract a descriptor, such as SIFT
4. Take a sample of the corpus of descriptors from the training set, and store these 'visual words'
5. Map each local descriptor to the matching visual words
6. Create a matrix consisting of pairwise similarities of the training images
7. Use this matrix to train an SVM

The recognition is then performed in these steps:

1. Detect features in the input image
2. Use the SVM to classify the image

2.5 Region proposals

R-CNN [Girshick et al., 2014] is an object detection method which combines Region Proposals with Convolutional Neural Networks (CNN). Several improvements to R-CNN have been given since the initial algorithm.

Two problems which needed to be solved for object detection methods which make use of CNNs, were the localization of objects, and dealing with scarcely labeled datasets. Several approaches for localization were used in the past, including regression, and sliding windows. There were certain problems with using the regression and sliding windows methods, as discussed in [Girshick et al., 2014]. R-CNN solves these problems by performing "recognition within regions". This method generates approximately 2000 region proposals for the input image, and then extracts a feature vector of all the candidate matches, which is classified using a linear Support Vector Machine (SVM).

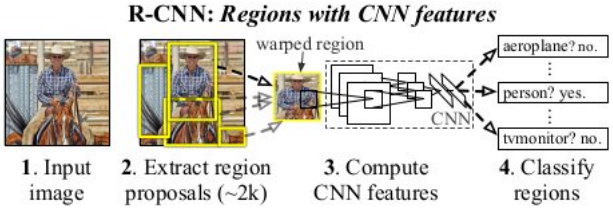


Figure 2: The R-CNN pipeline. Courtesy of [Girshick et al., 2014].

The problem of scarce datasets was conventionally addressed by first performing unsupervised pretraining, and then performing supervised fine-tuning. The R-CNN method, however, addresses this problem by first performing supervised training on a large auxiliary dataset(ImageNet Large Scale Visual Recognition Challenge [Deng, 2012]), and then performs domain-specific fine-tuning on a smaller dataset [Everingham et al., 2010]. R-CNN has an mAP (mean average precision) of 54% on the PASCAL VOC 2012 dataset, and detects objects with three *modules*:

1. The first module generates the *region proposals*, which defines the set of candidates.

2. The second module is a *CNN*, which creates a feature vector for each candidate

3. The third module is a linear *SVM*, used for classification of the feature vectors.

Building on the work from R-CNN, Fast R-CNN [Girshick, 2015] was proposed. It introduced several innovations (discussed in [Girshick, 2015]) which improved training and testing speed, and improved detection speed (9x faster) and accuracy. There were four major improvements of Fast R-CNN:

1. Firstly, it had a higher detection accuracy.

2. Secondly, the training was reduced to only a single stage, whereas the R-CNN training process was multistage.

3. Thirdly, it allowed updating the network layers.

4. And lastly, less disk storage was required during training.

Fast R-CNN achieved a mAP of 65.7% on the VOC12 dataset.

Faster R-CNN [Ren et al., 2015] further improved the algorithm, by introducing a *Region Proposal Network* (RPN). A RPN is a convolution network that is capable of predicting object bounds and scores at each position. Faster R-CNN has an mAP of up to 78.8% on the PASCAL VOC 2007 test set.

Mask R-CNN [He et al., 2017] is an extension of Faster R-CNN which is capable of generating segmentation masks.

A potential downside of using R-CNN is its complexity. The training involves training separate classifiers, which is slow and hard to optimize.

2.6 Single Shot MultiBox detector

Single Shot MultiBox detector (SSD) [Liu et al., 2016] is an object detection method which, unlike the multiple networks in R-CNN, uses only a single deep neural network. This makes the method simpler to train and use, while still achieving a mAP of 74.3% on the VOC2007 test set, at 59 frames per second. SSD outperforms Faster R-CNN by having a higher accuracy, and 3x the speed.

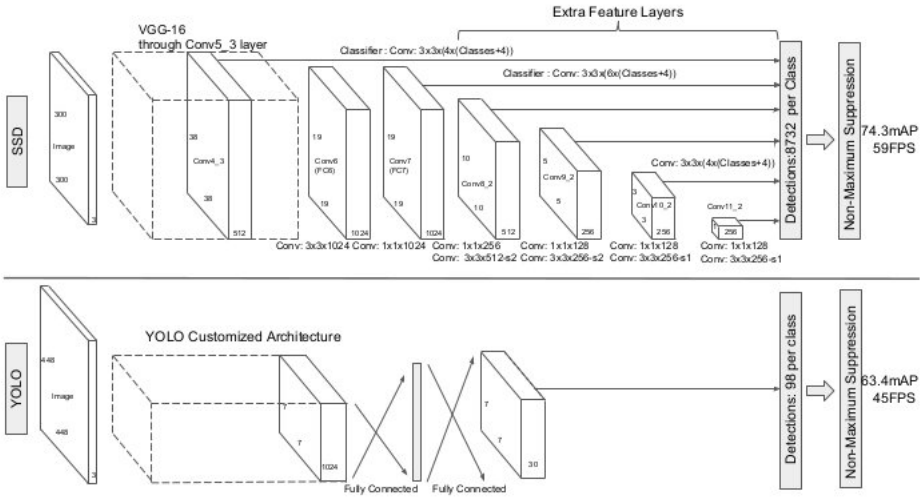


Figure 3: A comparison of YOLO and SSD. Courtesy of [Liu et al., 2016].

2.7 You Only Look Once

You Only Look Once (YOLO) [Redmon et al., 2016] is another extremely fast object detection method. Similar to SSD, YOLO does not perform object detection with multiple neural networks, but it uses only a single network to predict bounding boxes, and classes of objects. YOLO is capable of processing images at 45 frames per second, and a smaller version of the network can even process images at 155 frames per second, while still having a very high mAP.

A downside of YOLO is that it is less accurate than other state-of-the-art object detection systems. YOLO struggles to find the exact location of certain objects, and also with finding smaller objects.

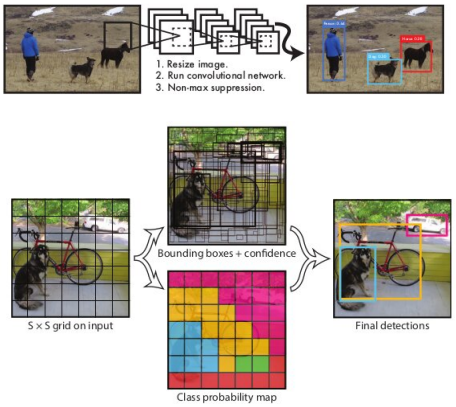


Figure 4: The YOLO algorithm. Courtesy of [Redmon et al., 2016].

3 Method

3.1 Object detection methods

Speed and accuracy are the main two selection criteria for the best occluded animal tracking method. As seen by the improvements of the mean accurate precision (mAP) (discussed in section 2), recent object detection methods, which make use of convolutional neural networks (e.g. R-CNN, YOLO, SSD), are more accurate than earlier methods such as Viola-Jones or SIFT. Since accuracy plays an important role in finding in real-time object detection, this research is mainly focused on the more accurate methods which make use of CNNs.

A second important part of real-time object detection, is speed. Of the recent methods which were compared, SSD and YOLO were the fastest and capable of detection at 59 frames per second, and 155 frames per second, respectively.

A downside of YOLO is that the algorithm struggles to find the location of smaller objects [Redmon et al., 2016]. Detecting smaller objects is an important requirement for finding objects from an aerial point of view. Since drones or unmanned aerial vehicles often fly at high altitudes, objects filmed at an aerial view, due to perspective,can appear smaller than their actual size. Therefore, YOLO might not be the best choice, unless the camera is close to the objects it is filming.

SSD and (Faster) R-CNN were capable of detecting smaller objects in our experiments. Of the two, SSD is more fast and accurate. However, Mask R-CNN, an extension of Faster R-CNN, has a feature which can draw an object mask around the objects inside bounding boxes. The segmentation mask can be useful for tasks such as predicting poses. It was therefore decided to make use of mask R-CNN for this research.

3.2 Experiments

In the experiments, we tested and measured how robust Mask R-CNN was with the detection of animals under occlusion. Due to time limitations, it was not possible to create training data with pictures of real dogs, therefore five stuffed animals of different shapes and sizes were used to perform the experiments. The stuffed animals were placed behind different objects, and pictures were taken with a Logitech Quickcam PRO 4000 webcam. Also, a video was taken of a dog, using a Motorola Moto G Smartphone.

Two sets of pictures of stuffed animals, and a video of the dog were taken. In the first set of pictures a total of 47 pictures were taken of stuffed animals on a plain green background from above. In the second set of pictures, a total of 40 pictures were taken of stuffed animals from above, which were hidden behind different types of objects. The video was taken of a black and tan Doberman Pinscher, on a grey floor, as the canine chased a rubber ball. The video consisted of a total of 352 frames. Figures 5, 6 and 7 show a sample of the images taken of stuffed animals, stuffed animals under occlusion, and the frames of the video from the dog, respectively.

Facebook AI Research software Detectron [Girshick et al., 2018], powered by the Caffe2 [2] deep learning framework was used for the implementation of

[2] `https://caffe2.ai/`

Mask R-CNN. The model used for recognition was a pretrained model trained on Microsoft's Common Objects in Context (COCO) dataset [Lin et al., 2014].

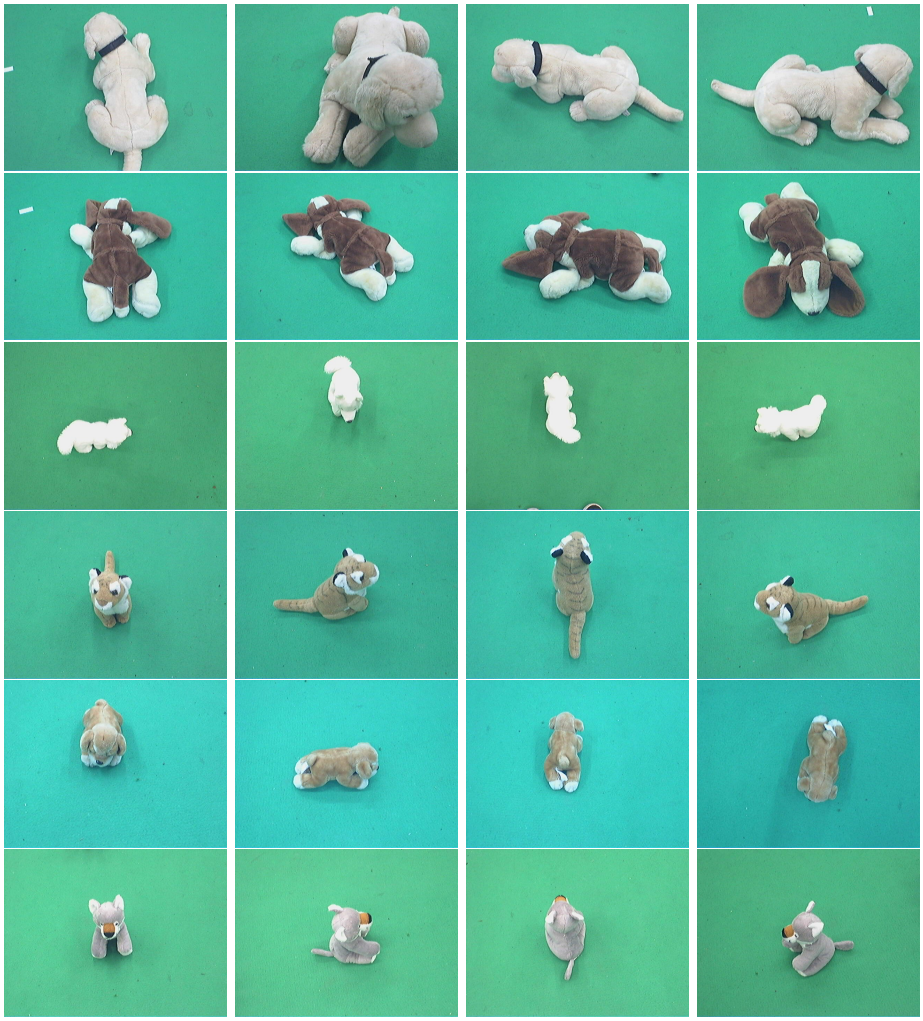


Figure 5: Pictures of stuffed animals taken without occlusion, using a Logitech Quickcam PRO 4000 webcam

Figure 6: Pictures of stuffed animals taken under occlusion, using a Logitech Quickcam PRO 4000 webcam

Figure 7: Some of the frames of a video captured using a Motorola Moto G smartphone. The frames were taken of a black and tan Doberman Pinscher as it chased a rubber ball.

4 Results

This section briefly discusses the results of the experiments in which two datasets of images from stuffed animals (with and without occlusion), and images from a dog were classified using the Mask R-CNN algorithm. The images were chosen as correctly classified if the classname (e.g. dog, or teddy bear) was correctly guessed, and the bounding box and segmentation mask were neatly fitted over the respective object.

Dataset	**Accuracy**	**Algorithm**
Stuffed animals (no occlusion)	80%	Mask R-CNN
Stuffed animals (with occlusion)	45%	Mask R-CNN
Frames of dog video (no occlusion)	80%	Mask R-CNN

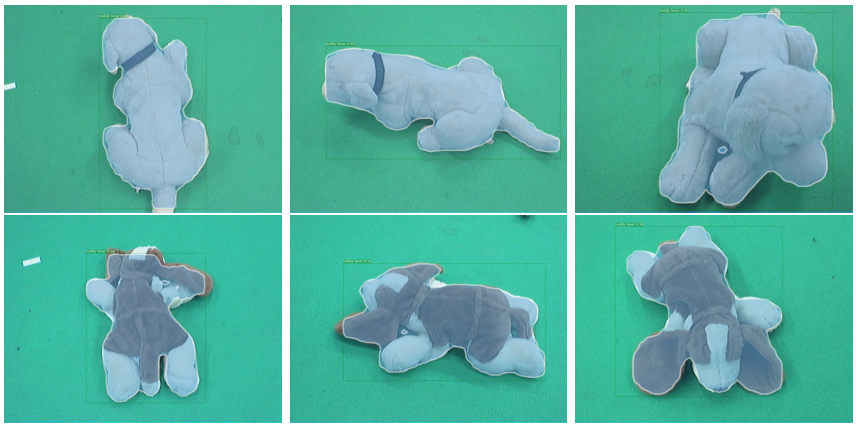
Table 1: The results of classifying three datasets using the Mask R-CNN algorithm.

4.1 Recognition of stuffed animals on plain background

Figure 8 shows a sample of the results of the first experiment, where 47 images were taken from stuffed animals. The stuffed animals were placed on a green floor, in a well lit room. The pictures were taken from above with a webcam.

In each subfigure of figure 8, a light green box can be seen, which is drawn over each detected stuffed animal. The predicted classes are shown in the top left of each bounding box. A light blue mask is drawn over each of the stuffed animals. All the stuffed animals resemble dogs, tigers, and zebras. The first four rows of figure 8 show stuffed animals which were correctly classified. The last row shows stuffed animals which were incorrectly classified.

38 of the 47 images were recognized, and classified as either a dog, a zebra, or a teddy bear. Since all the stuffed animals beared resemblance to dogs, zebras, or teddy bears, respectively, these classifications were counted as correct. There was thus an accuracy of 80%.



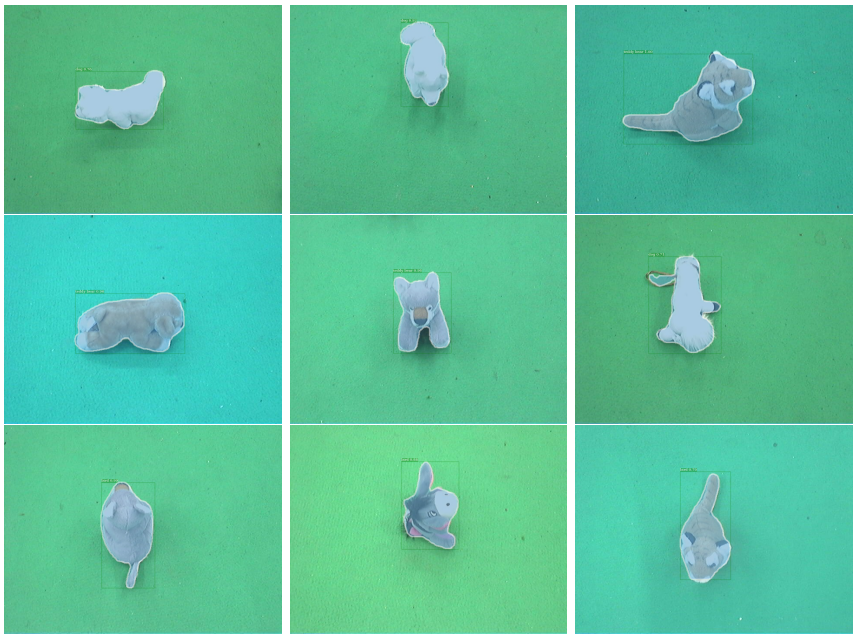


Figure 7: The output of the algorithm after classifying pictures of stuffed animals on a plain green background. A light green bounding box, and blue segmentation mask are drawn over each object. The predicted classes, from left to right, top to bottom are: teddy bear, teddy bear, teddy bear, teddy bear, teddy bear, teddy, bear, dog, dog, teddy bear, teddy bear, teddy bear, dog, bird, bird, bird

4.2 Recognition of stuffed animals under occlusion

Figure 9 shows the result of the second experiment, where stuffed animals were placed behind several objects, and pictures were taken from above, using a webcam. The objects behind which the stuffed animals were placed, were: a grey and white colored NAO robot, a white and blue scoreboard with the number 42 on it, a light brown cardboard box, and a small artificial Christmas tree. The objects were chosen to fit the scenario which was described in the challenge by SciRoc, on which this thesis is based. The scenario took place in a shopping mall, where grey and white objects, cardboard boxes, cards with lettering, and decorated trees, are common objects.

Of the 40 pictures of stuffed animals which were occluded behind different objects, 18 were classified as zebra, dog, or teddy bear. As with the previous experiment, we count these as correctly classified, since the animals did bear resemblance to zebras, dogs, and teddy bears, respectively. There was an accuracy of 45%.

The left column of figure 8 shows incorrect classifications, and the right column shows correctly classified stuffed animals. The first, second and fourth pictures, from top to bottom, in the left column show false positives. In these false positives, the NAO robots, behind which the stuffed animals were placed, were classified as teddy bear, but the stuffed animals behind it were not recognized. In the third picture of the left column, a cup, which is on top of the

Christmas tree is correctly classified as cup, which is irrelevant for this research. The white stuffed animal near the center, of the picture, however is not recognized. A possible explanation for this is because of the close proximity of the white colored stuffed animal, to the white underground next to which it is placed.

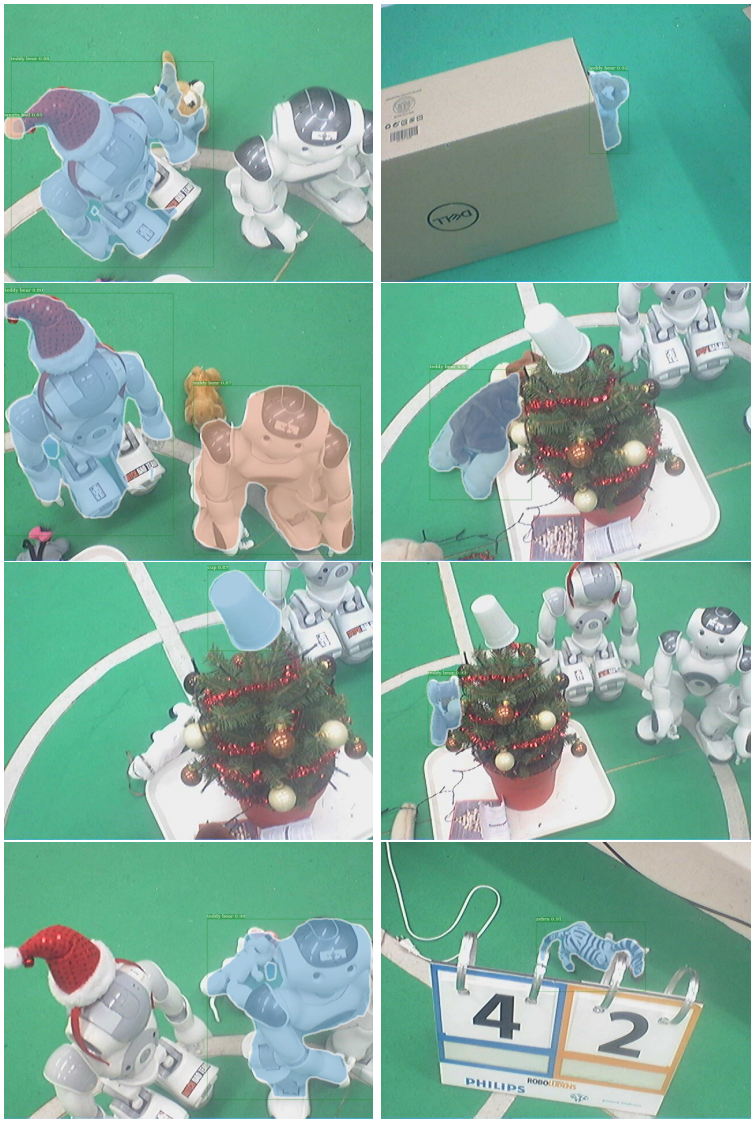


Figure 8: Pictures of stuffed animals taken under occlusion, using a Logitech Quickcam PRO 4000 webcam. The left column shows pictures which were incorrectly classified, the right column shows incorrect classifications. The first, second and fourth pictures, in the left column, from top to bottom, show false positives. In the third picture, a cup is recognized as cup. The white stuffed animal near the center of the picture is not recognized. The classes in the right column, from top to bottom are: teddy bear, teddy bear, teddy bear, zebra.

4.3 Recognition of of canine in video

The results of the last experiment are shown in figure 10. The left column in figure 10 shows the frames of the dog video which were incorrectly classified. The right column shows frames which were correctly classified. A light green bounding box is drawn over each recognized object, and a blue segmentation mask is used. The predicted classes are shown at the top right of the bounding box.

The dog was correctly classified in, 283 frames of the total 352 frames which were captured. There was thus an accuracy of 80%.

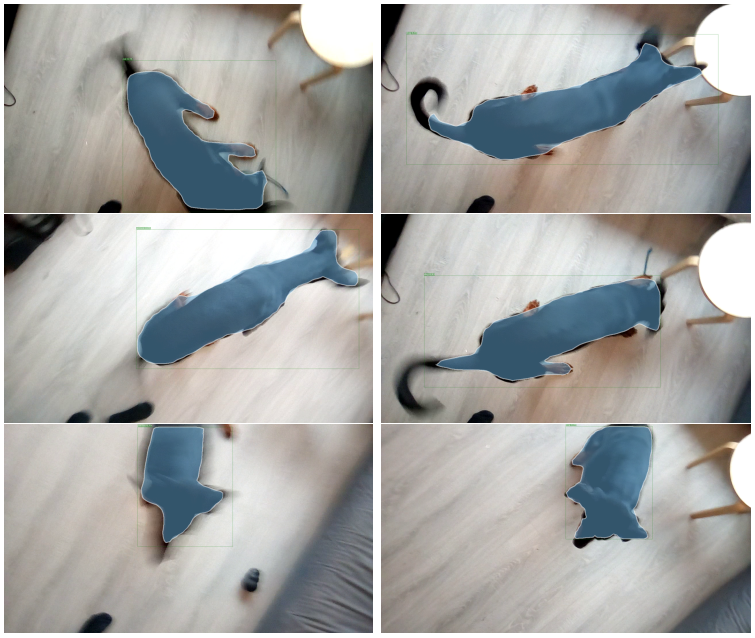


Figure 9: Some of the output frames of a video captured using a Motorola Moto G smartphone. The frames were taken of a black and tan Doberman Pinscher as it chased a rubber ball. The left column shows incorrect classifications, the right column shows correct classifications. The predicted classes in the left column, top to bottom: cat, person, person. The predicted, classes in the right column, top to bottom: dog, dog, dog

5 Discussion

With an accuracy of 80%, the images of stuffed animals on a plain green background had a relatively high accuracy. The high accuracy can be explained by the fact that the stuffed animals were clearly visible, as the room in which the pictures were taken was well lit. The images of the real dog also had an accuracy of 80%, but these images were also taken without any objects blocking the view. Unexpectedly, some of the pictures of dogs (Figure 10) were misclassified as being a person, instead of a dog.

The accuracy significantly dropped on the dataset of the images taken from stuffed animals under occlusion. It is suspected that the low accuracy is caused by the training set not having enough training examples of stuffed animals under occlusion. Figure 11 shows two examples of images from the training set. In both images, the teddy bears are clearly visible, and not partially hidden behind other objects. It is hypothesized that the accuracy could be improved with more images of stuffed animals under occlusion within the training set. The accuracy of the images from dogs can presumably also be improved by adding more images to the training set, which might avoid misclassifications such as classifying a dog as a person.

Figure 10: Pictures of teddy bears from the COCO dataset.

There are some notable misclassifications visible in figure 9, especially the first, and last images in the left column. One of the obstacles behind which the stuffed animals were placed, was a NAO robot. In the misclassifications, a teddy bear resembling a tiger, is placed behind the NAO robot. The NAO robot is wearing a red Christmas hat. After using the images as an input for the algorithm, the robot is incorrectly classified as a teddy bear. In this instance, the segmentation mask is drawn over the robot and slightly over the teddy bear in the back. Since we only counted a classification as correct if the bounding box and segmentation mask were neatly over the object itself, this image was counted as incorrectly classified.

In future research, it might be useful to use a more accurate measure of correctness, in order to prevent counting classifications (such as the above example) as incorrect, which still have some correctly guessed pixels. One such method could be to compute the ideal center of all the pixels in the object, measure the distance between the locations, and determine the center of the pixels within the algorithm's output. With this sort of approach, the aforementioned image could generate a higher level of accuracy as some pixels would be classified as correct.

6 Conclusion

Several real time object detection techniques were briefly discussed. Then an experiment was conducted to test how well the Mask R-CNN algorithm performed with the task of detecting occluded dogs (represented as stuffed animals), from an aerial point of view. Two sets of pictures and a video were taken. Both the pictures, and video were taken from an aerial point of view. The pictures were taken of stuffed animals, with one set of pictures under occlusion, and the other without occlusion. The video was taken of a moving dog.

The frames of the video, and the pictures without occlusion, both had an accuracy of 80%. The accuracy significantly dropped with the image set of stuffed animals under occlusion. It is suspected that the drop in accuracy is due to insufficient training data. Due to time constraints it was not possible to gather more training data.

The experiments which were conducted do suggest that Mask R-CNN is a suitable method for tracking objects. We hypothesize that, given enough training images of occluded animals from an aerial point of view, it should be possible to use Mask R-CNN to find them. For future research it is therefore suggested to gather more training data of stuffed animals, or dogs, from an aerial point of view, with the stuffed animals or dogs under occlusion. This research made use of 2D camera's. However, future research might benefit from 3D camera's, since the extra dimension might help with more accurately detecting objects.

References

[Csurka et al., 2004] Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague.

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE.

[Deng, 2012] Deng, J. (2012). Imagenet large scale visual recognition competition. `http://www.image-net.org/challenges/LSVRC/2012/`.

[Dollar et al., 2012] Dollar, P., Wojek, C., Schiele, B., and Perona, P. (2012). Pedestrian detection: An evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence*, 34(4):743–761.

[Everingham et al., 2010] Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338.

[Freund and Schapire, 1997] Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139.

[Girshick, 2015] Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448.

[Girshick et al., 2014] Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587.

[Girshick et al., 2018] Girshick, R., Radosavovic, I., Gkioxari, G., Dollár, P., and He, K. (2018). Detectron. `https://github.com/facebookresearch/detectron`.

[Grauman and Leibe, 2011] Grauman, K. and Leibe, B. (2011). Visual object recognition. *Synthesis lectures on artificial intelligence and machine learning*, 5(2):1–181.

[Hardin and Jensen, 2011] Hardin, P. J. and Jensen, R. R. (2011). Small-scale unmanned aerial vehicles in environmental remote sensing: Challenges and opportunities. *GIScience & Remote Sensing*, 48(1):99–111.

[He et al., 2017] He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE.

[Lin et al., 2014] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

[Linchant et al., 2015] Linchant, J., Lisein, J., Semeki, J., Lejeune, P., and Vermeulen, C. (2015). Are unmanned aircraft systems (uas s) the future of wildlife monitoring? a review of accomplishments and challenges. *Mammal Review*, 45(4):239–252.

[Liu et al., 2016] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

[Redmon et al., 2016] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788.

[Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

[Rodríguez-Canosa et al., 2012] Rodríguez-Canosa, G. R., Thomas, S., Del Cerro, J., Barrientos, A., and MacDonald, B. (2012). A real-time method to detect and track moving objects (datmo) from unmanned aerial vehicles (uavs) using a single camera. *Remote Sensing*, 4(4):1090–1111.

[Rowley et al., 1998] Rowley, H. A., Baluja, S., and Kanade, T. (1998). Neural network-based face detection. *IEEE Transactions on pattern analysis and machine intelligence*, 20(1):23–38.

[Viola and Jones, 2001] Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE.

[Viola and Jones, 2004] Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, 57(2):137–154.

[Yu et al., 2003] Yu, X., Xu, C., Leong, H. W., Tian, Q., Tang, Q., and Wan, K. W. (2003). Trajectory-based ball detection and tracking with applications to semantic analysis of broadcast soccer video. In *Proceedings of the eleventh ACM international conference on Multimedia*, pages 11–20. ACM.