Geo-information Science and Remote Sensing

Thesis Report GIRS-2018-25

Suitability Assessment of Object Trackers for Unmanned Aerial Vehicle video data

A study towards the performance of third-party tracking algorithms on a wildlife video dataset acquired by an Unmanned Aerial Vehicle

T.J.N. Jak

15th of August 2018

WAGENINGEN
UNIVERSITY & RESEARCH

Suitability Assessment of Object Trackers for Unmanned Aerial Vehicle video data

A study towards the performance of third-party tracking algorithms on a wildlife video dataset acquired by an Unmanned Aerial Vehicle

T.J.N. (Tim) Jak

Registration number 94 03 18 387 060

Supervisors:

Dr. D (Devis) Tuia
B.A. (Benjamin) Kellenberger

A thesis submitted in partial fulfilment of the degree of Master of Science
at Wageningen University & Research,
The Netherlands.

15th of August, 2018
Wageningen, The Netherlands

Thesis code number: GRS-80436
Thesis Report: GIRS-2018-25
Wageningen University and Research Centre
Laboratory of Geo-Information Science and Remote Sensing

Foreword

This research is part of the final assessment for the Master Geo-Information Science, part of the Laboratory Geo-Information Science and Remote Sensing. I chose this topic, because I was really interested in the application of non-invasive methods for wildlife tracking and I was challenged by the unknown field of computer vision.

I worked on this topic from November until August and during this period I experienced the help of several people. First of all, I would like to thank my supervisors, Devis Tuía and Benjamin Kellenberger, for their support and enthusiasm throughout the entire process. I am grateful for the time and effort you put in guiding me. I also would like to thank the SAVMAP consortium for providing the wildlife videos and the Wageningen University & Research for their people's guidance and a working place. At last, I would like to thank Jelle ten Harkel and Sophie Stuhler for proofreading this report and giving valuable feedback.

In case of questions about the developed KWT dataset or the joint detector-tracking system, please contact the author by the details provided by the cover.

Abstract

Understanding why and how animals move is essential for conservation purposes. Their movements provide insight in group dynamics, interaction or avoidance behaviour and habitat preferences that could be related to environmental variables, food availability and reproduction strategies. Animal tracking is often done using telemetry. The technology of telemetry is developing fast with increasing resolution, lighter transponders and more sampling points. However, it still has some limitations: the method still requires handlings on the animal, the collar is likely to affect the animal's behaviour and the transponders are relatively expensive. One of the techniques that offers the potential to aid in tracking animals is Unmanned Aerial Vehicle (UAV) based video imagery. It can cover remote areas and does not require any handlings on the animal. To further improve this technology, it is required to automate the process of detecting and tracking animals in the video datasets. There are already algorithms developed that can automatically find or follow an object, but they have only been tested on videos with humans or traffic as object of interest. It is expected that a wildlife dataset will cause different conditions, with for example shades and smaller, camouflaged animals. Therefore, this study will assess the performance of third-party object trackers and use the best tracker for a joint detector-tracking system. The trackers are assessed on a specifically developed video dataset, called the Kuzikus Wildlife Tracking (KWT) dataset. The dataset contains African wildlife captured from UAV perspective with remarkable characteristics like occlusion, similar objects, objects rotating or abruptly changing their direction and fast camera movement. The following five object trackers are assessed in this study: Kanade-Lucas-Tomasi (KLT), Kernelized Correlation Filter (KCF), Dual Correlation Filter (DCF), Tracking-Learning-Detection (TLD) and Least Soft-threshold Squares Tracker (LSST). The KCF and DCF outcompeted all other trackers, where KCF performed slightly better than DCF. It should be mentioned that KCF and DCF provided the possibility to run on Histogram of Oriented Gradients (HOG) features instead of raw image pixels that were used for the other three trackers. KCF and DCF performed especially well on characteristics like similar objects and objects changing their appearance, but failed in occlusion events and abrupt change of direction. KLT on raw image pixels did succeed in tracking objects that suddenly changed their direction, but failed on most other characteristics. At last, TLD failed on almost all characteristics and was often not able to re-detect objects after occlusion. It can be concluded that KCF performed best on the KWT dataset. Therefore, this tracker is combined with an object detector. This detector can re-initialise the tracker in case it lost the object of interest. Surprisingly, the detector-tracking system performed worse than KCF on HOG features, especially when similar objects were present in the same video. On the other hand, the new system was able to handle occlusion events by using a linear interpolation method.

Keywords: *Object tracking, Tracking-by-detection, UAV imagery, Benchmark dataset, African wildlife*

1 Introduction

Understanding why and how animals move is essential for conservation purposes (Kays et al., 2015; Millspaugh & Marzluff, 2001; Spiegel et al., 2017). It provides insight into the animal's habitat preferences, which can be related to environmental variables, food availability and reproduction strategies (Aarts et al., 2008; Beyer et al., 2010). In case of tracking multiple individuals, movements could explain inter- and intraspecific interactions between animals such as attraction and avoidance behaviour (Kays et al., 2015; Li et al., 2013; Long et al., 2014), where avoidance behaviour towards human objects might reveal animal disturbances and increased stress (Ellenberg et al., 2013; A. Wilson et al., 2015). Animal movements also reveal group dynamics within communities, such as dominance towards other individuals (Benhamou et al., 2014; Dey & Quinn, 2014). To know where animals are, why they are there and following them to understand their behaviour becomes essential when dealing with conservation and management issues in, for example, habitat loss or fragmentation (Brudvig et al., 2015). Moreover, it may also help in preventing human-wildlife conflicts or traffic collisions (Manohar et al., 2018). Therefore, several methods have been developed to track animal positions. For example, camera traps, artificial tags (e.g. bird rings), natural marks (e.g. fluke and fin pattern of whales) and geolocators (Acevedo et al., 2017; Gill et al., 2014; Rowcliffe et al., 2016; Stanley et al., 2015). However, most animal tracking is done by telemetry (Kays et al., 2015; Millspaugh & Marzluff, 2001).

One of the first studies that made use of radio-telemetry to track animals was conducted in the 1960s (Cochran, 1963). The animals were relatively heavy, so they could deal with the weight of the transmitter, and the sampled area covered only a few hectares. A lot of development has been made since that moment. Transponders have become more accurate, smaller and lighter in weight and they can measure for longer time periods (Dewhirst et al., 2016; Rutz & Hays, 2009; Wikelski et al., 2007). This resulted in a larger variety of animals that could be tracked, because researchers aim to keep the weight of transponders under 5% of the animal's body weight (Boitani & Fuller, 2000; Wikelski et al., 2007). Recently, it is even possible to measure physiological and environmental parameters, like heart-beat rate, body temperature and acceleration, all of which provide insight into the behaviour of the animal (Rutz & Hays, 2009; A. Wilson et al., 2015). From this perspective, it might seem that telemetry is a valid monitoring technique to track wildlife and record behavioural activity. However, also this method imposts some limitations: the process of providing animals with a transponder can be seen as an invasive handling, as contact with the animal is required (Cooke et al., 2013; R. P. Wilson & McMahon, 2006). The transponder might affect the animal's behaviour (Aarts et al., 2008; White & Garrott, 2012), and around 65% of all mammal species cannot be tracked due to the 5% body-weight rule (Bridge et al., 2011; Kays et al., 2015). Furthermore, GPS trackers often fail in dense forests, causing habitats to be under sampled or not sampled at all (Hunter, 2007). Additionally, the transmitters are relatively expensive, which consequently results in lower sample sizes (Aebischer et al., 1993). Therefore, there is a need for different methodologies that overcome the limitations of telemetry.

One of the recent methodologies that may have the potentials to aid in monitoring animals is Unmanned Aerial Vehicle (UAV) based video imagery (Gonzalez et al., 2016; Grémillet et al., 2012). UAVs have been applied in several monitoring programmes on terrestrial and marine mammals (Bevan et al., 2015; Chabot & Bird, 2012; Ditmer et al., 2015; Hodgson et al., 2013; Rey et al., 2017; Vermeulen et al., 2013). The benefit of using UAVs is that they can access remote areas and that no direct interference with the animals is required (Lucieer et al., 2014). The reduction of human-wildlife interactions likely contributes to the expression of more natural animal behaviour and thus more reliable measurements (White & Garrott, 2012).

At this moment, videos acquired by UAV are processed manually to detect, count or track animals of interest. Manually going through the videos is a time-consuming task and prone to human mistakes.

In some cases immediate video analysis is required, for example when poachers are a threat for the animal of interest (Mulero-Pázmány et al., 2014). Therefore, it is essential to automate this process of detection and tracking for efficient deployment of UAVs as a monitoring tool (van Gemert et al., 2014). There are already automated tracking algorithms that determine the trajectory of the object of interest throughout a video sequence (Borji et al., 2015; Wu et al., 2015). However, most of them are only tested on videos with human faces, pedestrians or vehicles as object, but not on videos of wildlife recorded from UAV perspective. It is expected that this will cause different conditions for the tracking algorithms, with for example shades and smaller, camouflaged objects. Therefore, this research will test the performance of third party trackers on a wildlife video dataset, and develop an automated wildlife monitoring system based on the best performing tracker.

A wide range of tracking algorithms are available that use different methods to determine object position. For example, the trackers proposed by Hong et al. (2015) and Hare et al. (2016) that use a Convolutional Neural Network and Support Vector Machine respectively to locate the object of interest. Those supervised methods belong to the more sophisticated methods in the field of object tracking. The term supervised refers to the process of providing annotated images to the tracker, which can be used to train the classifier to determine whether it is the object of interest or not. This means that beforehand a lot of images with different representations of the object are required, which is often not the case in the field of wildlife tracking. Therefore, this research focuses only on unsupervised trackers that do not need an extensive dataset with annotated images to train their selected labels. The only input required for those trackers are the coordinates of the object of interest (i.e. the animal) in the first frame. As a second criterion, the trackers should perform well on characteristics that are common in wildlife videos, like object or camera rotation, occlusion, illumination or similar objects (i.e. same animal species). Most benchmarks that assess the performance of trackers indicate this performance per characteristic (Smeulders et al., 2014; Wu et al., 2013, 2015). Based on the above mentioned criteria, five trackers have been selected: Kanade-Lucas-Tomasi (KLT; Tomasi and Kanade 1991, Shi 1994), Kernelized Correlation Filter (KCF) and Dual Correlation Filter (DCF; Henriques et al. 2015), Tracking-Learning-Detection (TLD; Kalal et al. 2012) and Least Soft-threshold Squares Tracker (LSST; Wang et al. 2013).

The performance of those trackers will be assessed on a video dataset called the Kuzikus Wildlife Tracking (KWT) dataset, that has been created for this study (chapter 4). The KWT dataset contains UAV-based video footage with oblique and top-down views on various African ungulates and other large mammals. Each video consists of some specific characteristics and it is expected that some trackers perform better on certain characteristics than others. Therefore, the trackers will also be assessed on their performance on the following characteristics: occlusion, similar objects, changing object direction, camera movement and object rotation (e.g. changing object appearance). This should answer the following research questions:

1. Which proposed tracker is most suitable for tracking wildlife, tested on the KWT dataset?
2. How do the selected trackers perform under characteristics like occlusion, similar objects, changing object direction, camera movement and object rotation?

To develop an automated wildlife monitoring system, the best assessed tracker will be combined with an object detector for optimal performance. This detector works on a Convolutional Neural Network (CNN) and is developed by Kellenberger et al. (2017). The idea is that the detector can reinforce the tracker in two ways: by initialising the tracker in the first frame and by re-initialising the tracker in case it lost the object of interest. In the end, the performance of this joint detector-tracking system will be assessed on the KWT dataset to answer the last research question:

3. What is the performance of the detector-tracker monitoring system on the KWT dataset?

Those three research questions contribute towards the objective of this study: Find the best object tracker for wildlife tracking and combine it with the detector of Kellenberger et al. (2017) to contribute towards a wildlife monitoring tool. The contributions of this study are three-fold:

- The KWT test dataset with 34 annotated, wildlife objects.
- The objective comparison and discussion of the most suitable tracker out of the selection for tracking wildlife objects from UAV perspective.
- An automated wildlife monitoring system based on the combination of the object detector of Kellenberger et al. (2017) and the best assessed third party tracker.

This report provides an overview of the steps that have been taken, the results that have been retrieved and an explanation of the results based on what is found in literature. Chapter 2 introduces the topic of automated wildlife tracking using object trackers by reviewing related literature. The next chapter provides a short description of the used trackers in this study. In chapter 4 the characteristics of the KWT test dataset are described. All trackers are run on this dataset and assessed on their performance. Several metrics have been developed for this assessment, which are described in chapter 5, including the ones used in this research. The analysis steps and the used software can also be found in this chapter. The object detector and the principles behind the joint detector-tracking system are described in chapter 6, including some additional features that could be examined in further research. Chapter 7 presents the results, divided in the overall performance metrics and the detailed performance on the specified characteristics. In chapter 8 I provide a discussion on the results. Chapter 9 summarises the results by answering the research questions and the last chapter gives some recommendations for additional studies.

2 Related studies

Several studies on automated wildlife tracking using video data have been performed already, all focusing on a different domain of species (Dell et al., 2014). For example, on counting and tracking fish (Spampinato et al., 2008), detecting and tracking individual cattle (Andrew et al., 2017), or movement and behavioural studies on captive insects (Branson et al., 2009). This chapter will highlight three different studies to give an impression of what is possible in automated wildlife video tracking. The three studies focus on wild terrestrial mammals and they use different techniques to detect and track the animals throughout a video.

Burghardt and Ćalić (2006) used the face detection algorithm of Viola and Jones (2004) and combined this with the KLT tracker (section 3.2). The detection algorithm was trained on lion faces and, when the faces were visible, tracked throughout the video sequence. The method was used to classify simple behaviour like walking, trotting or standing. One of the requirements was that the lions had to look into the camera to be detected or tracked, something that is normally not the case in wildlife videos. However, by using the strongly textured faces and a large training dataset of almost 1700 images, they achieved a high number of true positives (i.e. correct predictions). A different technique was proposed by Zeppelzauer (2013), who used a smaller dataset of 10 to 20 images to train a classifier. The system was developed to detect and track elephants, which are in some cases similar to their background colour. Nevertheless, the method used colour segmentation to detect homogeneous segments of the elephant. The object segments were tracked over their neighbouring frames and matched with the predicted segments within those frames. If the segments intersected with each other, a link was created. This link was then used to determine whether a segment is a true positive or a false negative (the tracker fails to find the object). It is considered that long links, without too many gaps are true positives and false negatives otherwise. The algorithm also takes shape and texture into account to improve the decision of whether segments are an elephant or not. True positives are characterised by slowly changing shapes and low texture. Both systems of Burghardt and Ćalić (2006) and Zeppelzauer (2013) only focused on detecting and tracking one single species. Manohar et al. (2018) proposed a system that was able to detect, track and identify a larger group of objects. The objects were separated from the background, and thus detected, using a *maximum similarity-based region merging algorithm*. This algorithm requires the image to be segmented into homogeneous regions, for example by using mean shift segmentation (Ning et al., 2010). The next step is to merge and classify the small, homogeneous regions into the object of interest or background. The merging process is based on a similarity value between one region and its neighbouring regions. Two regions are only merged together if they have the highest similarity value compared to the other neighbours. Beforehand, it is known for some regions whether it is the object of interest or background. This knowledge can be used to classify other, unknown regions as well. The final result is an object region that is separated from the background, which is used as input for the *mean shift-based tracking algorithm*. In here, the object is represented by a colour histogram. In the next frame, this colour histogram is checked against a template region, which is a region in the image based on the object's previous location and its certainty (Yang et al., 2005). At last the objects are identified to species level using a *k-nearest neighbour classifier*.

The above mentioned techniques were developed to track large animals that were most of the times clearly visible in the videos. According to my knowledge, only the study of Risse et al. (2017) focused on small animals as well, with the specifically created Wildlife Animal Tracking (WAT) dataset to test their tracker on. Their objects ranged in size from a larger coyote to a small woodlouse of only a few pixels. However, most videos were not taken from UAV perspective and the dataset did not contain any long-term video sequences. Therefore, my study is unique in having a dataset that contains videos taken from UAV perspective with a wide range of African wildlife in their natural environment.

3 Working Principle of the Evaluated Trackers

3.1 The principle of object tracking

Object tracking is an important part in the computer vision domain, with a lot of new trackers developed every year (Čehovin et al., 2016; Yilmaz et al., 2006). It can be defined as *following the trajectory of an object throughout a sequence of consecutive video frames* (Kalal et al., 2012; Smeulders et al., 2014) and serves many applications like surveillance (Ojha & Sakhare, 2015), traffic monitoring, vehicle navigation (Yilmaz et al., 2006), cell biology (Sbalzarini & Koumoutsakos, 2005) and professional sports (Buchheit et al., 2014). Object tracking differs from *object detection* in the sense that trackers follow the object throughout the whole video sequence and use information from those previous frames to determine the object trajectory (Kalal et al., 2012). Object detectors locate the object in a single image and often work in a supervised manner. They are often used to find the object of interest, which serve as initialisation of the object trackers (see examples in chapter 2). The focus of this research will only be on object trackers that serve the purpose of long-term (TLD) and short-term tracking (KLT, KCF, DCF and LSST), where long-term tracking is defined as the task of processing at frame-rate, indefinitely long (Kalal et al., 2012).

In general, trackers consist of five components: a motion model, feature extractor, observation model, model updater and sometimes an ensemble post-processor (cf. Figure 1) (N. Wang et al., 2015). In the first frame the object of interest has to be annotated, either manually or by a detector. This representation of the object is called the "base patch". In consecutive frames this base patch is compared with "candidate patches" that are generated by the motion model of the tracker of interest. This model takes the location and feature representation of the object in the previous frame into account to propose candidate patches that could be the object of interest. The feature extractor represents the base patch and each candidate patch by its features, which can be defined as characteristics that describe the object of interest. Often used features are colour, grayscale (N. Wang et al., 2015; Yilmaz et al., 2006), Haar-like features (Viola & Jones, 2001) or Histogram of Oriented Gradients (HOG) features (Dalal & Triggs, 2005). Based on those features, the observation model decides whether the candidate patch is the object of interest. It calculates a similarity or probability value for this and the patch with the highest probability is selected as the object of interest of the current frame. The model updater decides whether the observation model needs to be updated, i.e. if the new object representation needs to be added to the collection of base patches. It needs to find a balance in updating and it has to be sure that it is the object of interest so that it does not take wrong object representations into account, which would amplify errors. The last component, the ensemble post-processor, is quite specific and is not applied on every tracker. It can be used to improve the tracking result by running multiple trackers simultaneously and take the ensemble of the output bounding boxes. This results in a more stable final bounding box.

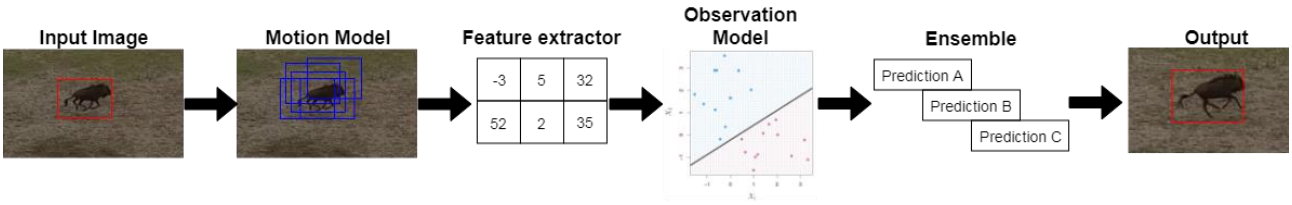


Figure 1: Simple representation of the general components in tracking. Please refer to James et al. (2013) for the image of the observation model.

All trackers function in a similar way as described above by N. Wang et al. (2015). However, not every tracker uses the same strategy. Trackers can be subdivided in generative or discriminative trackers (N. Wang et al., 2015). Generative trackers focus on patches that show the highest similarity values with

the object of interest. They build an appearance model of the object and search for a similar representation in the relevant frame (Asha & Narasimhadhan, 2017). An example is the LSST tracker used in this research of D. Wang et al. (2013). On the other hand, discriminative trackers train their classifier to exclude the object of interest from the background. Those type of trackers often outperform generative trackers, because they can better handle complicated backgrounds. Trackers can also differ in object representation. A distinction can be made between feature point tracking, silhouette tracking and kernel-based tracking (Joshi & Thakore, 2012; Yilmaz et al., 2006). In feature point tracking the detected object of interest consists of several points with for example high textured areas or corners, as in the KLT tracker (section 3.2). Those points are tracked throughout the sequence and together they indicate the position of the object. Silhouette and kernel-based trackers both make use of the inside information of the object, like colour, texture and sharp edges between the object and its background (Joshi & Thakore, 2012). This information is used to propose and select the best candidate patches per frame. Those candidate patches are often close to the object location in the previous frame, due to the use of a motion model (Santner et al., 2010). In this study, three of the five trackers (KCF, DCF and TLD) are considered as kernel-based trackers.

To understand how the proposed trackers are able to track the object, or why certain trackers lose the object, a brief description of the characteristics per tracker is given. Please refer to the original papers of the trackers for a more elaborated explanation.

3.2 KLT

The Kanade-Lucas-Tomasi (KLT) tracker is the oldest and conceptually the most simple tracker used in this research. Figure 2 shows the general working principle of KLT. It selects small windows (cyan dots) within the bounding box of interest, selected in the initial frame (red rectangle). The small windows contain characteristic features of the object that are then used to track those windows throughout the video sequence (Shi, 1994; Tomasi & Kanade, 1991). Additionally, a new bounding box is drawn around the outer feature windows for every frame (yellow rectangle). This bounding box represents the object's area and is calculated by forward geometric transformation between the old and new points.

To understand the working principle of the KLT tracker, it is necessary to describe what defines characteristic feature windows, how the algorithm selects them and how it tracks the windows from frame to frame. In the initial frame suitable feature windows should be selected that are identical and distinguishable from background clutter, which is described as the background near the target having the same colour or texture as the object (Wu et al., 2013). The feature windows are automatically selected by the algorithm, based on regions that are rich in texture or regions that have a high spatial frequency content (Tomasi & Kanade, 1991). Examples of good feature windows are object corners or areas with strong contrasts in image intensities, as described in Shi (1994). After the feature windows have been selected in the initial frame, they are tracked throughout the other frames. KLT does this frame by frame, where the new object location *J(x)* is determined by the old object location from the previous frame *I(x)* and the displacement *(d)*:

$$J(x) = I(x - d) + n(x) \qquad (1)$$

here, *n(x)* is the residual noise that could not be explained by the displacement. The displacement is the amount of movement of a feature window between the two consecutive frames, *I(x)* and *J(x)*. It is chosen to minimize the dissimilarity, defined:

$$\epsilon = \int_W [I(x - d) - J(x)]^2 \; w \, dx \qquad (2)$$

where *w* is a weighting factor. The dissimilarity will increase in case a feature window changes in appearance relative to the windows in previous frames, for example caused by illumination, object rotation or occlusion. The KLT tracker monitors this dissimilarity value in two ways. The first method calculates the dissimilarity value between the current and previous frame by looking at the translation in the image space. This method is used for the tracking process, where the lowest dissimilarity value determines the displacement between two consecutive frames. The second method calculates the dissimilarity between the current and first frame, using affine motion (Shi, 1994). This value is used to determine the performance of a feature window, which will be discarded in case it exceeds a predefined threshold. Figure 2 shows the selected feature windows in the initial frame (left) and after 10 frames (right). In this example, the feature windows around the object are lost, because they showed too large dissimilarity values. On the other hand, the windows encompassing the object itself are preserved. The number of tracking windows within the object's bounding box determine the tracking performance. In case too many windows have been lost, the tracking process stops without the possibility to re-detect the object.

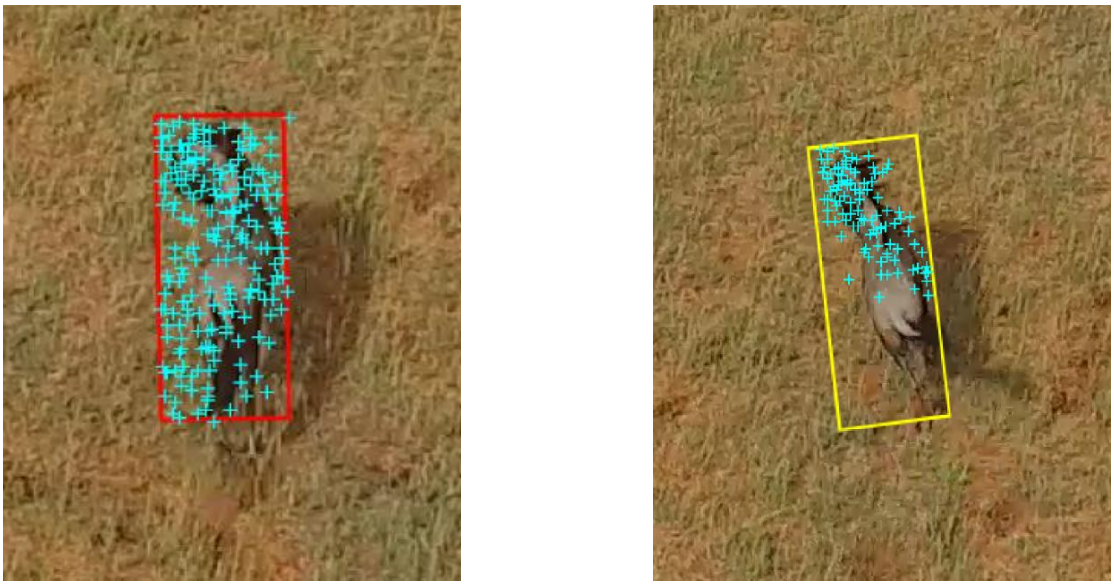


Figure 2: The feature windows calculated by the KLT tracking algorithm indicated as cyan dots in the initial frame (left) and in frame 10 (right). The red bounding box depicts the object of interest, selected by the user. The yellow bounding box represents the tracked object area.

3.3 KCF and DCF

The Kernelized Correlation Filter (KCF) and Dual Correlation Filter (DCF) are two similar trackers, only differentiated by the kernel type. They were proposed by Henriques et al. (2015), where they performed similar to state-of-the-art trackers. Therefore, both trackers are included in this research to evaluate their performance on the KWT dataset.

KCF and DCF are kernel-based trackers with the object of interest represented by a rectangular bounding box. In the first frame the object of interest is initialised as the base patch. The window around this object is a bit larger to provide some context (Henriques et al., 2015). A classifier to distinguish the object from its background is on this base sample (positive examples) and on translations of the base sample (negative examples). Henriques et al. (2015) uses ridge regression as classifier, as it was found to be efficient and to yield accuracies similar to common models like Support Vector Machines (SVM) (Weston et al., 2001). Both trackers use cross-correlation to detect the object in the subsequent frames, which works in a sliding window fashion that can vary in size. The window moves over the whole image and decides for every possible location whether the underlying patch

contains the object of interest or not (Henriques et al., 2015). To speed up the process of comparing the base window over the image, the correlation is computed in the Fourier domain. A classifier decides which proposed candidate patch corresponds most with the base patch and thus is likely to be the object of interest. This classifier makes use of a linear ridge regression kernel in case of the DCF tracker, while KCF uses a non-linear Gaussian kernel. The linear kernel of DCF applies the 'kernel trick', which means that every point is mapped in a higher dimensional space (Henriques et al., 2015). In complex situations, this allows to obtain a non-linear response using only linear operations (cf. Figure 3) (Cortes & Vapnik, 1995). In most cases both kernels return similar outputs and it is expected that KCF and DCF will be similar in tracking result (Henriques et al., 2015).

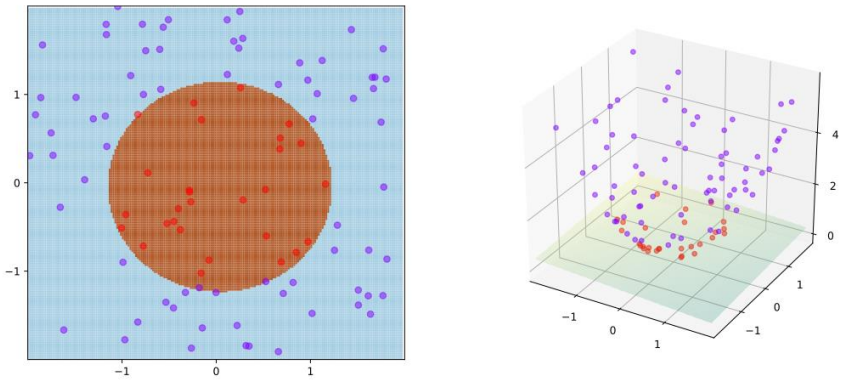


Figure 3: Example of a Gaussian kernel in a 2D and 3D representation. In the 3D representation the points can be classified using a linear operation (Picture from Ji (2016)).

To improve the performance of both trackers, more advanced features can be used as inputs, like Histogram of Oriented Gradients (HOG) (Barbu, 2014; Danelljan et al., 2015; Tang et al., 2007). HOG features are characterised by gradients or edges and work by creating a histogram based on the gradient direction and magnitude (Dalal & Triggs, 2005; Geismann & Schneider, 2008; Mallick, 2016). This way, the base sample histogram can be compared with the histograms of the sliding window that sampled over the image. Its effectiveness and improved performance is shown by Henriques et al. (2015) and therefore both KCF and DCF will be run and assessed on HOG features.

3.4 TLD

The Tracking, Learning and Detection (TLD) tracker is developed by Kalal et al. (2012). The tracking component follows the object throughout the video and provides training data to the detector. The detector learns the observed appearances and corrects the tracker if necessary, while the learning component estimates detector errors and updates it (Kalal et al., 2012). This integrated principle makes the tracker suitable for long-term tracking and allows it to re-detect objects after occlusion. Several studies acknowledge this already (Henriques et al., 2015; Smeulders et al., 2014) and it will be interesting to see whether the tracker is able to re-detect the object of interest, especially considering similar objects in the same frame.

As with all investigated trackers, TLD must be initialised in the first frame. However, the detecting component expects labelled input data to train its classifier. To overcome this problem, the learning component trains the classifier part in the detector based on the object that is denoted by the user. This object is considered as a positive example in the labelled data, while all patches around it are negative examples. Additional positive examples are generated by selecting 10 bounding boxes as close as possible around the initial object, which are then transformed (shifted, scale change, in-plane rotation)

20 times per bounding box. This results in 200 positive examples, that are slightly transformed compared to the initial object. The detector then uses this labelled data to find the object in the next frames. To do so, it uses the same sliding window approach as the KCF and DCF tracker, but only on relevant areas, selected according to a threefold procedure: first, all patches with a gray-value variance smaller than 50% are excluded. This discards more than half of the patches, that mostly contain background information. In the second stage an ensemble of base classifiers is applied on the remaining patches. The number of base classifiers is obtained from the gathered positive examples so far and should represent the object of interest. Each base classifier performs a number of pixel comparisons with the candidate patch, which results in a probability. The candidate patch goes to the last stage if the mean probability is above 50%. In the last stage, a nearest neighbour classifier is applied to the positive object detections. In the end this results in one or several bounding boxes that i) contain the object of interest, ii) missed the object of interest or iii) found another object.

The learning component initialises the detector and updates it using P-N learning (Kalal et al., 2012), based on the growing and pruning principle of Kalal et al. (2009). The P-expert identifies only false negatives and the N-expert only false positives (the tracker finds an object, that is actually background). This means that the P-expert accounts for appearance variations of the current object. Therefore, it uses the temporal structure of the video and expects that the object is moving along a certain trajectory. If the position predicted by the P-expert is not overlapping with the detected location, then the location is still added as a positive example to the ensemble classifier. On the other hand, the N-expert tries to find clutter in the background by using the spatial structure. It takes the predictions of the detector and tracker into account and calculates which detection is most confident. The most confident detection is used to re-initialise the tracker, while the detections that are not overlapping with the most confident one are considered as negative examples (i.e. background).

The last part of the TLD tracker is the tracking component. This component only needs the position of the object in the first frame and works in an unsupervised manner. It is based on the Median-Flow tracker (Kalal et al., 2010), which uses the movement of specific points between two consecutive frames. The specific points are determined by a pyramidal version of KLT (Bouguet, 2001). The movement of the most reliable points determine the predicted location in the next frame. By checking the residual between individual points and the median of all points, failures or occlusion events might be found. In case this residual exceeds a certain pixel threshold, no bounding box is returned. In that case, it is up to the detector to re-detect the object. In this way the tracker and detector can reinforce each other's capabilities.

3.5 LSST

The Least Soft-threshold Squares Tracker (LSST) will be assessed due to its performance on partial occlusion events, where it outperformed TLD (D. Wang et al., 2013). LSST also showed good performance on challenges like fast motion, illumination and background clutter, which are particularly prevalent in the KWT dataset.

In contrast to the other four trackers, LSST is a generative tracker. This means that it is creating a series of patches representing the object of interest, here called a dictionary, and try to match that with given candidate or target patches. To find the most likely target patch per frame, the tracker makes use of linear regression. It fits a linear line (red dotted line) through a number of base observations which are certainly representations of the object of interest (red circles; Figure 4). This line is fitted in such a way that it keeps the residual or error term as low as possible, where it assumes that those residuals are independent and identically distributed. D. Wang et al. (2013) modelled the residuals in an innovative way as Gaussian and Laplacian noise, called a Gaussian-Laplacian distribution. In here, the Gaussian component models small noise and the Laplacian component models outliers (e.g. occlusion events). By making use of this Gaussian-Laplacian distribution, they made sure that the red dotted line

was not influenced by outlying observations (blue circle). A comparable method that also excludes outlying observations is robust regression with a Huber loss function (Huber, 1964). This method gives a lower weight factor to outlying residuals, so that the calculated model parameters are less influenced by them.

In every frame new candidate patches (green squares; Figure 4) are proposed that could be the object of interest. The most likely candidate patch is selected based on a distance value that is inversely proportional to the likelihood that a patch is the object of interest (D. Wang et al., 2013). So with an increase in the likelihood, the distance value will decrease. The candidate patch with the lowest distance value is thus most likely the patch with the object of interest. After finding the most likely candidate patch, the dictionary with base patches is updated. It could be that the most likely candidate patch is still an outlying value compared to the red dotted line. In that case, the Laplacian noise term detects it as an outlier and replaces it by an average value. In case it is not an outlier, the candidate patch with its original values is added to the other base patches and used in the next comparisons. This way the tracker can deal with object changes due to, for example, illumination.

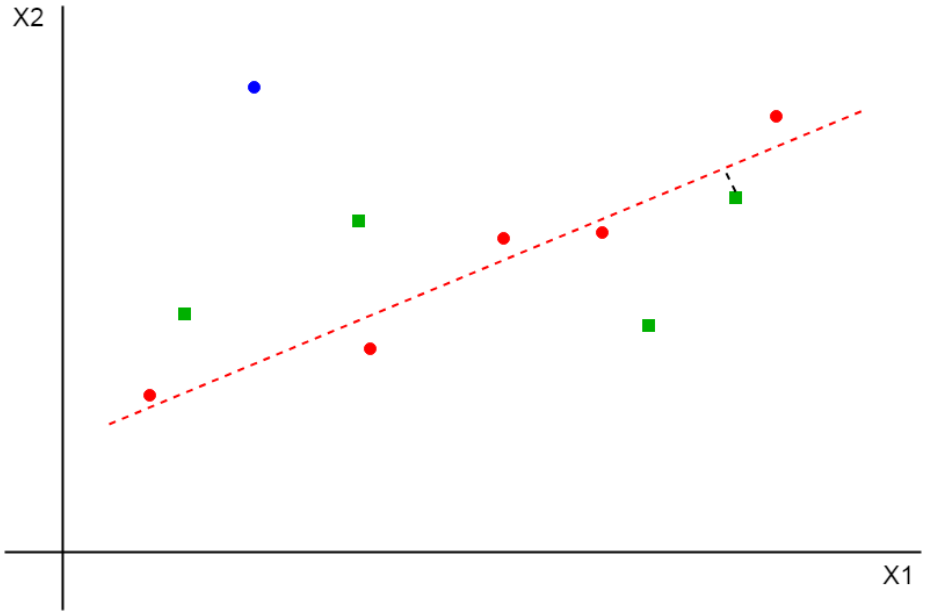


Figure 4: A fictional illustration of the decisive process of LSST. The circles represent base observations, of which the blue one is an outlier. The dictionary is fitted through the red circles and decides, based on a distance value, whether a target patch is the object.

4 The Kuzikus Wildlife Tracking dataset

The KWT dataset contains wildlife in their natural habitat acquired by a UAV. The videos in the original dataset were recorded as part of a project by the SAVMAP Consortium[1]. Their goal is to develop a wildlife monitoring tool by using ultrahigh-resolution images (LASIG, 2016). The videos were acquired in the Kuzikus Wildlife Reserve park[2] in Namibia between 12 and 15th of May 2014. Out of this dataset several subsamples have been taken to serve as a test dataset, including specific characteristics like occlusion, similar objects, changing object direction and object or camera rotation. The videos in the KWT dataset have a resolution of 3840 x 2160 pixels and vary in length from 2 until 53 seconds, or 81 until 1594 frames respectively. A combination of a letter and a number is used to refer to specific videos. The letter indicates the video, while the number defines the particular object in that video. In total there are 18 videos with 34 objects to track. This means that in some videos multiple objects have been tracked. The smallest object contains on average 249 pixels (±16 x 16 pixels) and the largest object 35621 pixels (±189 x 189 pixels). The minimum average object velocity is 2.36 pixels per frame with a maximum of 20.88 pixels. Please refer to Appendix I for an impression and elaborate description of the dataset.

4.1 Manual Annotation and Quality Control

The KWT dataset is mainly developed for testing whether the predicted tracker locations match with the actual object location. To create those ground truth bounding boxes I used the Video Annotation Tool from Irvine California (VATIC) (Vondrick et al., 2013). This tool allows to manually draw bounding boxes around the object of interest and returns per frame the outer coordinates of the bounding box. Additionally, this tool was able to derive the individual frames per video, which were used to run the trackers on. The ground truth bounding boxes contain the complete animal body, including legs, tail and some background. The VATIC tool divided each video in segments of 320 frames, from which the last 20 frames were overlapping with the next segment. This allowed for some internal validation to see how accurate the ground truth bounding boxes had been drawn. Therefore, the Root Mean Squared Error (RMSE) between the centres of both bounding boxes had been calculated for three groups of small (500 observations), intermediate (500) and large objects (521). The small objects had an RMSE of 2.58 pixels, intermediate 3.12 pixels and large 4.29 pixels. To put this number into perspective, small objects had an average surface area of 1000 pixels, which comes down to a size of 31 x 31 pixels. Intermediate objects had an average surface area of 2950 (54 x 54) pixels and large objects 16150 (127 x 127) pixels.

4.2 Explanation of characteristics and perspectives

The KWT dataset contains five specific characteristics that are considered as important in wildlife tracking:

- Occlusion
- Similar species
- Change of object direction
- Object rotation
- Camera movement

An *occlusion* event is defined as an object not visible anymore, because it is hidden behind a static object like a tree. In case the object is hidden behind another specimen, it is not counted as an occlusion event, because it is too difficult for a tracker to make this distinction. An occlusion event is different from an *out-of-frame* event in the sense that the object comes back in the video after it occluded. Out-

[1] https://lasig.epfl.ch/savmap
[2] https://kuzikus-namibia.de/xe_index.html

of-frame events are only considered at the end of the video when the object is not coming back in the frame anymore. *Similar species* is another characteristic and defined as specimen of the same species present in the video. This was the case in most of the videos, because most animals were recorded in large herds. This characteristic is of specific interest in case the object has to be re-detected. The characteristic *change of object direction* can be interpreted in different ways. Here it is defined as an object changing its direction with an angle of more than 90 degrees within a maximum of 30 frames. The change of direction also causes a change in object appearance, or representation, meaning that it faces a different side than before. It is questionable whether the tracker can deal with such a vast change in the object's appearance, especially because it is trained on a different side of the object. A similar characteristic is *object rotation*, which is defined as the object changing its appearance, and thus showing different sides, throughout the video. It is different from changing object direction in the sense that there is a gradual instead of an abrupt change in appearance. The last characteristic, *fast camera movement*, occurred in only two videos (four occasions). It can be described as controlled movement of the camera to keep the object within the frame. This causes motion blur and the object to 'move' fast from one side in the image frame to the other side, which could result in losing the object.

In UAV video footage it should be considered whether the objects have been recorded from above (top-down view) or aside (oblique view) (Figure 5). It is expected that this can make a difference in tracking performance. Objects tracked from the top-down perspective are only recorded from one side. They can come closer to the camera or rotate, but their appearance stays more or less the same. However, with an oblique perspective the object can have several appearances, for example, by changing its orientation towards the camera. Due to such significant appearance changes, it is expected that objects are harder to follow from the oblique perspective, compared to top-down. Additionally, with the top-down perspective the UAV can reach higher altitudes which causes less animal disturbance and it makes it easier to keep the object in the centre of the video. However, only 4 of the 18 videos are taken from top-down perspective. Therefore, it is not possible to make a statistically sound comparison between both perspectives.

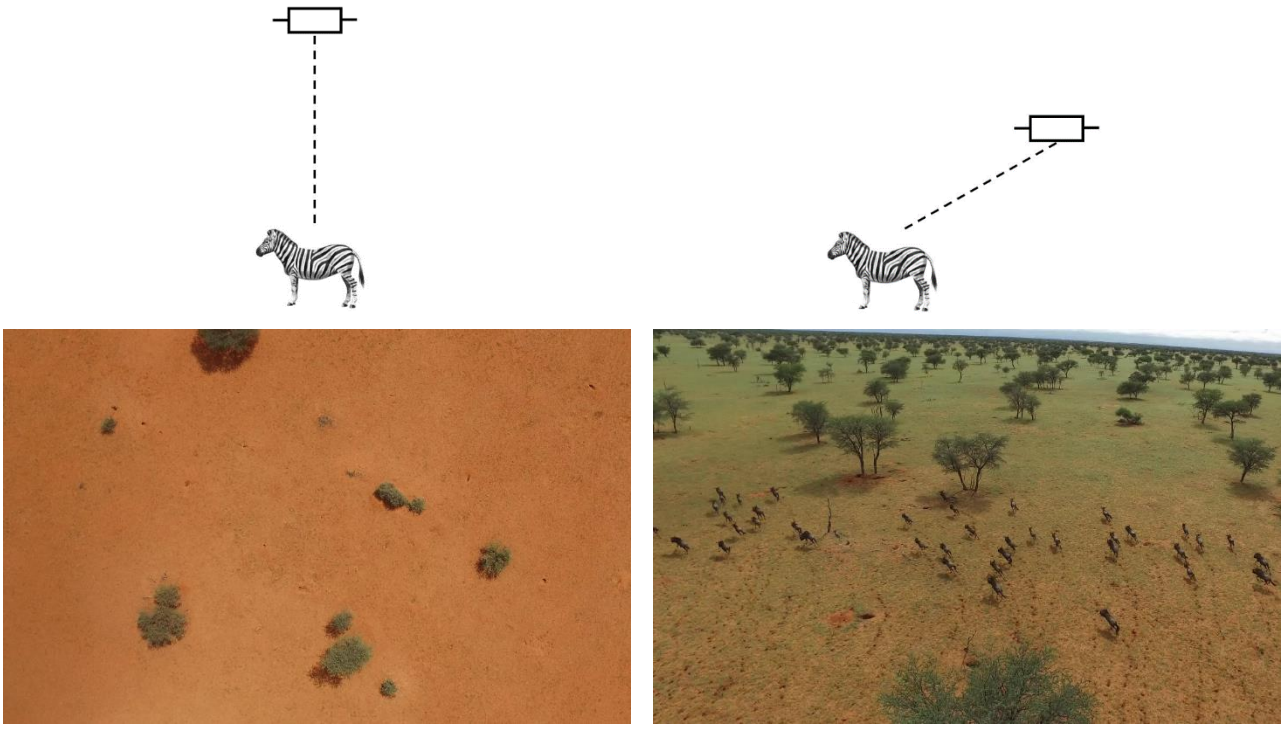


Figure 5: Visualization of top-down (left) and oblique (right) perspective with an example image from the KWT dataset.

5 Methodology

5.1 Evaluation metrics

A variety of metrics exist to assess and compare the trackers' performance, such as the general metrics as precision and success plot (Wu et al., 2015), robustness indicators (Kristan et al., 2015) and the F-score (Smeulders et al., 2014). Some metrics focus more on visualising the performance, for example the performance plots proposed by Čehovin et al. (2016). All metrics aim to depict a different aspect of the trackers' performance and the choice of an evaluation metric should thus be dependent on the problem definition. Therefore, this section describes the most common metrics that are found in object tracking literature. Section 5.2 explains the metrics used in this study and why they are selected.

Precision and success plot

Two widely used metrics are the precision and success plot. The precision plot, or centre location error plot, calculates the Euclidean distance between the centres of the ground truth and predicted bounding box (Figure 6A). A frame is considered as correctly tracked if this distance value is lower than a manually defined threshold. In here, the number of correctly tracked frames over the total number of frames is called the precision: $(N_{tp} / (N_{tp} + N_{fp}))$, where N_{tp} is the number of true positives per video and N_{fp} the number of false positives. The precision value is higher for larger threshold values. This interaction between the precision for certain threshold values is visualized in the precision plot (Figure 8). It shows clearly whether a tracker was able to follow the object or not, but it does not take the object size or rotation of the predicted object into account (Wu et al., 2015). The success plot does take the object size into account (Figure 6B). It makes use of the Intersection-over-Union metric, in which the overlapping area (grey area) is divided by the total area of both bounding boxes (red + grey + blue area). This results in an overlap score between 0 (no overlap) and 1 (complete overlap). The overlap score is used to calculate the success rate, e.g. the number of correctly tracked frames. The success plot also shows the success rate per threshold value. This success rate will be higher for low thresholds, which is opposite to the precision value. Since the precision or success rate could differ between trackers per threshold value, the Area-Under-Curve (AUC) value is used to rank the trackers according to their performance (Wu et al., 2013). The higher the AUC value, the better the performance of a tracker. The plots give a clear overview of the performance per tracker, but the main disadvantage is that they do not indicate what caused the error.

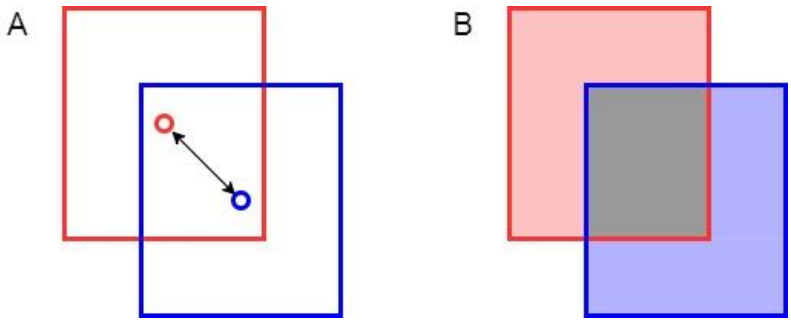


Figure 6: Sketch of a ground truth bounding box (red) and the predicted bounding box (blue) to explain the method of precision (A) and success plots (B).

Accuracy and robustness indicators

The accuracy and robustness indicators are often used in the Visual Object Tracking (VOT) challenges (Kristan et al., 2016; Kristan et al., 2015; Kristan et al., 2013). The accuracy measures the overlap between the ground truth and predicted bounding box, which in theory is the same as the success plot. Additional insights might be provided by the robustness, or failure count, which is a score of how

many times the tracker lost the object (Čehovin et al., 2016). An object is considered as lost in case the overlap or centre pixel value exceeds a certain threshold. The tracker will be externally re-initialised after it loses the object. The advantages of this method are that it deals with lost object events, that the entire sequence is taken into account and that there is less focus on the beginning of the sequence. However, such an evaluation is also time-intensive and in case the threshold value is changed, all videos need to be assessed again.

F-score

Smeulders et al. (2014) proposes the F-score as an evaluation metric for single object tracking. It needs a threshold to determine whether the tracker provides a true positive, false positive or false negative. A commonly used criterion for a bounding box to be a true positive is if its Intersection-over-Union with the ground truth bounding box exceeds 0.5 (Everingham et al., 2010). Besides the overlap, a pixel threshold can be used. The precision and recall (N_{tp} / (N_{tp} + N_{fn})) are calculated, where N_{fn} are the number of false negatives. The precision and recall are used to calculate the F-score, with higher F-scores for better tracking performances:

$$F = 2 * \frac{precision * recall}{(precision + recall)} \quad (3)$$

Tracking length and complete-tracking-performance

The tracking length is proposed by Čehovin et al. (2016). It measures the number of frames that are tracked correctly until the frame where the tracker lost the object. Bounding boxes can be counted as correctly tracked when they are within the boundary of a certain pixel or overlap threshold. Once this threshold is exceeded, counting is stopped, even if the object is re-detected in a later stage. Closely related to this principle is the complete-tracking-performance, which describes the number of videos that are tracked correctly until the end. To avoid false positives or false negatives due to partially occluded animals, this study considers the fifth-but-last frame as the end. The complete-tracking-performance gives insight in what kind of videos, and thus characteristics, a tracker can handle.

Performance plots

All metrics that are mentioned so far only indicate the overall performance of each tracker, without showing why a tracker loses the object. This information can tell much more about the functioning of a tracker and which characteristics cause failure. A possible answer to this is the performance plot. It visualises the performance of all trackers per video, with on the x-axis the frame number and on the y-axis the pixel distance. Since the objects differ in size and thus a fixed pixel threshold value is less informative in that case, it is advised to use a pixel distance relative to the object's size. Risse et al. (2017) proposed an *object length pixel distance*, calculated by dividing the pixel distance through the diameter of the object's bounding box. If this value is lower than 1, it indicates overlapping bounding boxes, while a value of 1 or higher indicates no-overlapping bounding boxes. This normalized value also gives more insight in how far a bounding box drifted away from the object. For the graphs a fixed value of 5 object length pixel distances is applied, to ensure that the focus of the graph is on the correct trackers.

5.2 Analysis

All tracking algorithms were open source and implemented in Matlab. The first step was to set up the environment and provide the frames to the trackers to run the algorithm on. The next step was the initialisation of the trackers. To avoid bias towards a specific tracker, the default settings of the trackers and the coordinates of the annotated ground truth of the first frame were used. Those coordinates were the same for all trackers. The last step, the evaluation of the trackers, consisted of three parts: 1) The

overall performance of all trackers, 2) TLD on occlusion and out-of-frame events and 3) specific performance plots for all trackers on specific video characteristics.

For the overall performance the Euclidean distance was calculated between the centre of the ground truth and the bounding box predicted by the trackers. A frame was considered as a true positive if this Euclidean distance value did not exceed the respective threshold. For the precision plot this threshold value ranged from 1 to 50, with more true positives for a higher threshold value. The precision was calculated per threshold value, by dividing the number of frames with true positives over the number of total frames. Since four of the five trackers were not designed to handle out-of-frame or occlusion events, those events were excluded from the precision plot (part 1). To evaluate the performance of the trackers over all threshold values, an additional Area Under Curve (AUC) value was calculated.

The F-score and complete-tracking-performance were also used to assess the overall performance per tracker. The F-score was calculated over the precision and recall parameters, which were derived from the number of true positives, false positives and false negatives. If a value was within the threshold of 50 pixels it was considered as a true positive. For the assessment of the complete-tracking-performance, the object length pixel distance was used to determine whether a tracker failed or not. It was calculated by dividing the pixel distance over the object length, which was considered as the diameter of the longest side of the bounding box. A tracker fulfilled the video in case it did not pass the threshold of one body length at any moment and made it until the fifth-but-last frame. Within those last frames the object was expected to be partially occluded, causing failure in the tracking performance. Additionally, it was noted if a tracker made it until the first occlusion in the video, since not all trackers were developed for occlusion or out-of-frame events. The success plot and accuracy index were not desired, since KCF and DCF did not update the size of their bounding box throughout the video and TLD used a fixed aspect ratio. The tracking length was also not included in the analysis, because it did not allow for re-detections made by TLD.

Part 2 of the analysis gave insight in the capabilities of TLD to handle occlusion and out-of-frame events. The behaviour of the TLD was visualised by a performance plot with additionally three images to show the event itself. Only a selection of all the videos in the KWT dataset was used to visualise the behaviour. This selection was based on interesting occasions, such as failures, or on the other hand, a re-detection.

The last part showed the performance of all the trackers, including TLD, on the characteristics occlusion, similar species, changing object direction, camera movement and object rotation. The performance of the trackers was visualised in the same way as in part 2, with the focus on videos where it was clear why a tracker failed or succeeded. It was expected that KCF, DCF, KLT and LSST could not deal with out-of-frame events, but short occlusions in turn might be feasible. Therefore, all videos in the last part contained no out-of-frame events and all trackers, including TLD, were tested on occlusion events as well.

5.3 Software

The annotation process was performed in VATIC (section 4.1.). KCF, DCF, KLT and LSST were all run on Matlab version R2015b. OpenCV[3] was necessary for running TLD, which was installed on a different system with Matlab version R2017b. Open source software like Python 2.7 and R (version 3.4.0) was used for calculating the Euclidean distances and creating the graphs and images of the performance plots.

[3] https://opencv.org

6 Detector-Tracking system

6.1 The object detector

The best assessed tracker will be combined with a detector developed by Kellenberger et al. (2017). Their detector is specifically designed to detect large animals from UAV perspective. The detector works on a pre-trained CNN called AlexNet (Krizhevsky et al., 2012), which is adapted with some additional layers by Kellenberger et al. (2017). The CNN makes use of a so-called two branch strategy. The first branch focuses on the recognition of potential objects by creating a probability map. The second branch uses the confidence score from the first branch to estimate the height and width of the object. As a result, the detector provides multiple object locations with a corresponding probability score. However, this probability score was not reliable to use, since in some cases false positives had a high score or true positives had a low probability score. The detector is not trained on the KWT dataset. The training dataset also consisted of wild African grazers, but only from top-down perspectives.

6.2 Implementation

The principle of a joint detector-tracking system is that the detector can reinforce the trackers performance. It is known that KCF, DCF, KLT and LSST cannot handle occlusions or out-of-frame events and therefore they are not suitable for long-term tracking. The detector could re-initialise these trackers in case they lose the object. On the other hand, the tracker can also improve the detector's performance. For example, by providing additional representations of the object during the tracking process, as done in TLD (Kalal et al., 2012), or by excluding false positives made by the detector (rocks, trees or other animals). The tracker can exclude those objects by following them over, for example, five frames and match the tracked bounding boxes with the detections. It is likely that the object of interest is detected again after five frames, while most of the other objects are not re-detected. This means that the tracked bounding box of a true positive overlaps with a detection, while most false positives are not overlapping anymore. By repeating this process over more frames, all false positives are excluded.

In the proposed system I only focused on re-initialising the tracker and excluding false positive detections. The detector-tracking system had to be manually initialised, because the detector did not provide the required outer bounding box coordinates. It only provided the centre coordinates of each detected object. After initialisation, the tracker's bounding box was matched with the locations predicted by the detector for every frame. It was considered that the tracker lost the object if the centre of its bounding box did not overlap with one of the predicted locations for 5 consecutive frames (red dotted boxes in Figure 7). In that case, the re-initialisation process was started. The first step in this process was to retrieve the last three correctly predicted bounding boxes (green dotted boxes). These three positions were used to calculate the direction, the speed of the animal (in pixels) and the parameters of a linear function. The direction only indicated whether the animal moved to the right or left side of the image and was calculated as follows: if the x-coordinate of the position on t-15 was smaller than the x-coordinate on t-5, than the object moved to the right (Figure 7). The object moved to the left if this was the other way around. Together with the animal's speed, this gave some potential x-coordinates of the animal's potential location. The margins of potential locations were extended by 10 pixels in case the animal moved faster or slower as calculated. For each potential x-coordinate, the corresponding y-coordinate was calculated by using the linear equation. This resulted in several potential locations, which were matched with the predictions of the detector. It was assumed that the closest distance between a potential location and a detection would most likely be the actual location. If the distance between this detection and the potential location was also within the margins of the object's bounding box, then the tracker was re-initialised with the detector coordinates.

The new detector-tracker system was also tested on the KWT dataset and compared with the best assessed tracker. The precision of both trackers was visualised per occasion using a line graph, with on the x-axis the occasion and on the y-axis the precision. Each occasion was subdivided into one of the characteristics to see on what kind of events the new system improved or decreased in performance. Beforehand, it was expected that the new system would perform better on occlusion events, while it would perform equal on the other characteristics.

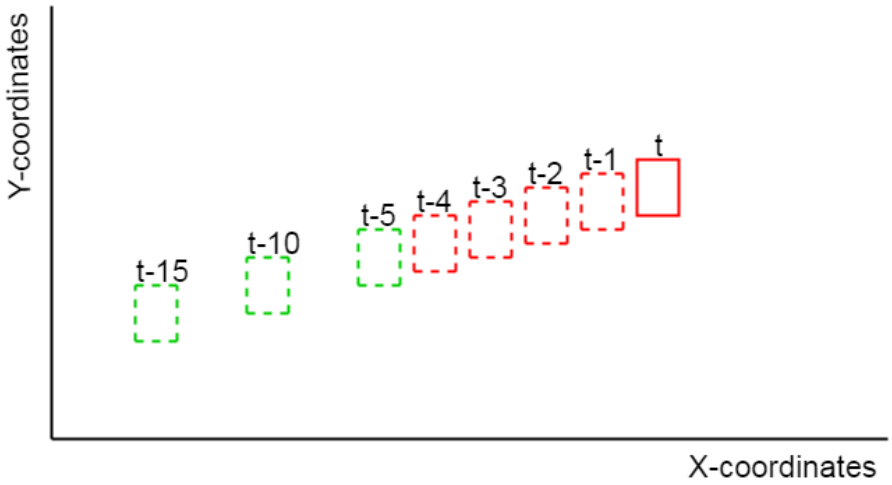


Figure 7: Representation of an image to visualise the process of re-initialisation. The tracker lost the object on time t, because it did not match with a detection for five consecutive frames (t until t-4). The green dotted bounding boxes represent the last three correctly predicted bounding boxes, which are used for predicting the animal's direction, speed and a linear algorithm to predict the corresponding y-coordinate.

7 Results

7.1 Overall performance

The overall performance of the five trackers is visualised in the precision plot in Figure 8. As mentioned before, the occlusion and out-of-frame events are excluded from calculating the precision in this graph. The additional AUC scores are used to rank the trackers and made it easier to compare among each other. The KCF and DCF tracker showed comparable results, with KCF performing slightly better with an AUC score of 12.76. Both trackers outperformed LSST (8.62), KLT (6.98) and TLD (4.93). Those AUC scores calculated without occlusion events did not differ that much from the scores that were calculated with occlusion events: KCF (12.47), DCF (12.45), LSST (8.43), KLT (6.82) and TLD (4.92). The general trend of KCF and DCF outperforming other trackers can also be seen in the F-scores (Table 1). The last measure to assess the overall performance is the complete-tracking-performance. KLT tracked 8 videos until the end. Four of the eight videos were taken from top-down perspective (A1, A2, B1, B2; see Appendix I). The videos were characterised by objects drastically changing direction (B2, J1) or similar objects (M1-M3). It was also the only tracker that was able to track the small, dark object in video C1 until it occluded. KCF and DCF tracked both the same six videos until the end. They were not able to process B2 entirely, where the object changed direction. However, they did perform well on videos D4 and D5 that include small, similar objects. They also fulfilled video K1 with a rotating object of interest, surrounded by similar objects. Videos O1 and O2, with both a similar object, and Q1 were tracked until the object occluded. LSST and TLD tracked three objects correctly, which were also completely tracked by KLT, KCF or DCF.

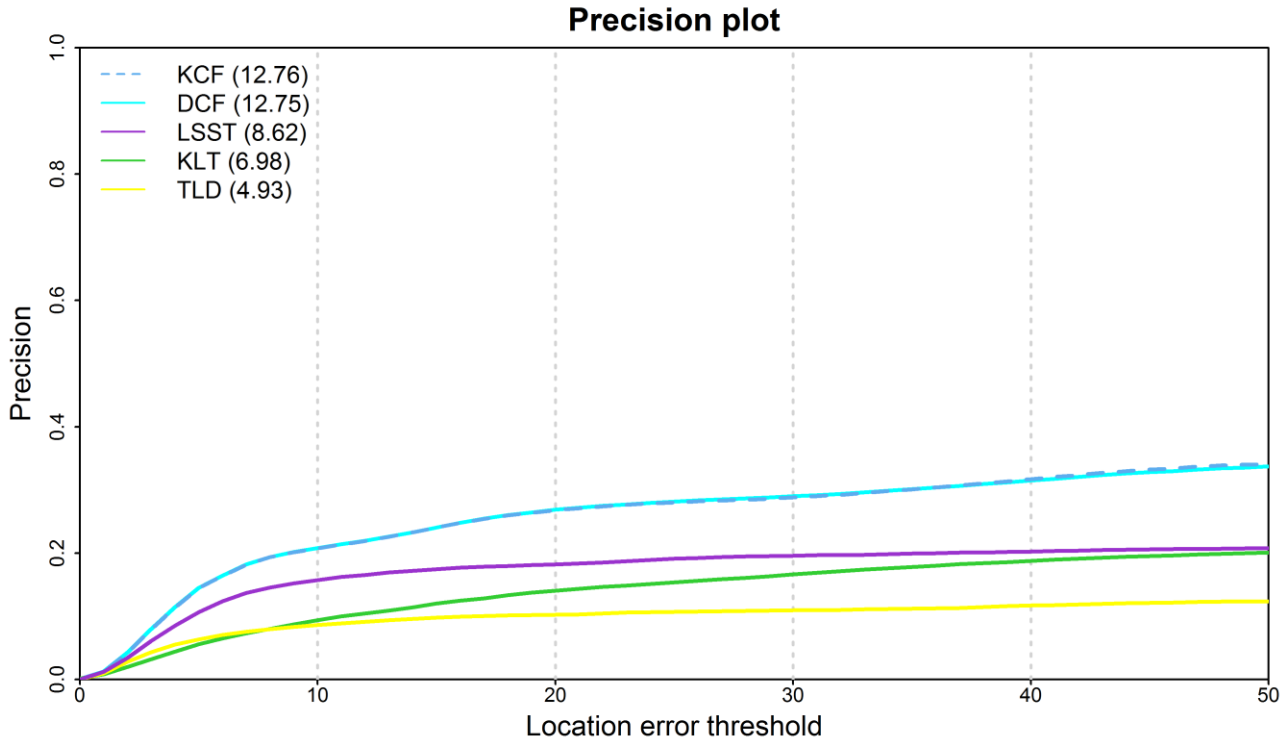


Figure 8: Precision plot for the five trackers with corresponding AUC values (with 100 as maximum AUC value). The precision is calculated by dividing the number of true positives by the total number of frames where the object is visible, i.e. without occlusion and out-of-frame events.

Table 1: Results of the F-score and the complete-tracking-performance per tracker. The F-score is calculated without frames with occlusion or out-of-frame events.

	F-score	Videos fulfilled	Videos fulfilled until occlusion
KCF	0.507	A1 - A2 - B1 - D4 - D5 - K1	L1 - O1 - O2 - P2 - Q1
DCF	0.502	A1 - A2 - B1 - D4 - D5 - K1	L1 - O1 - O2 - P2 - Q1
LSST	0.344	A1 - A2 - D4	L1 - O2 - P2
KLT	0.330	A1 - A2 - B1 - B2 - J1 - M1 - M3 - M4	C1 - L1 - P2
TLD	0.130	B1 - B2 - M4	L1

7.2 Capabilities TLD

Out-of-Frame

Video B1 (Figure 9) shows that TLD manages to notify that the animal is not present anymore in the video. It stopped tracking and did not provide any coordinates after frame 127. The images under the graph show the situation for three frames of interest. At frame 100 the animal already approached the boundary of the image and partially occluded in frame 125. In frame 127 the animal was completely out-of-frame, which was also noticed by TLD. The predicted bounding box overlapped almost perfectly with the ground truth. However, it can be noticed that the TLD was not completely accurate, because in this case it also tracked the shadow of the object.

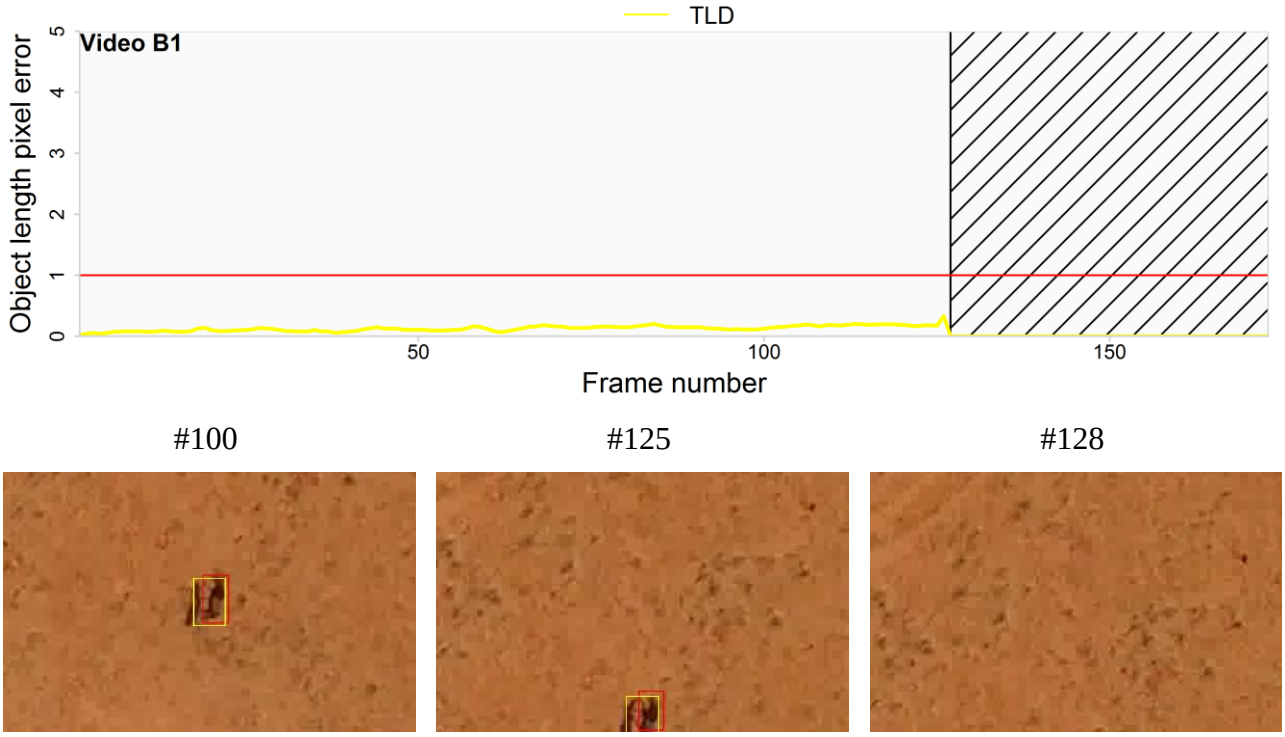


Figure 9: Performance plot of TLD on an out-of-frame event in video B1. The red line in the graph indicates the threshold of 1 object length and the black striped area indicates an out-of-frame event. Shown in the images are ground truth bounding boxes coloured in red and the predicted bounding boxes by TLD in yellow.

Occlusion

Occlusion events happened in 15 of the 34 occasions where an object was tracked. In video L1 (Figure 10) TLD noticed the animal as not present in the case it got occluded, which was considered as a true positive. The video only contained one animal that was occluded twice. The tracker lost the animal in frame 27, even if the object was not fully occluded yet. This is interesting to see, because video B1 (Figure 9) showed that TLD is able to keep tracking partially occluded animals. After the occlusion event, TLD was not able to directly re-detect the animal again. At frame 80 and 95 TLD found the object and tracked it in both cases for four frames after which it lost the object again.

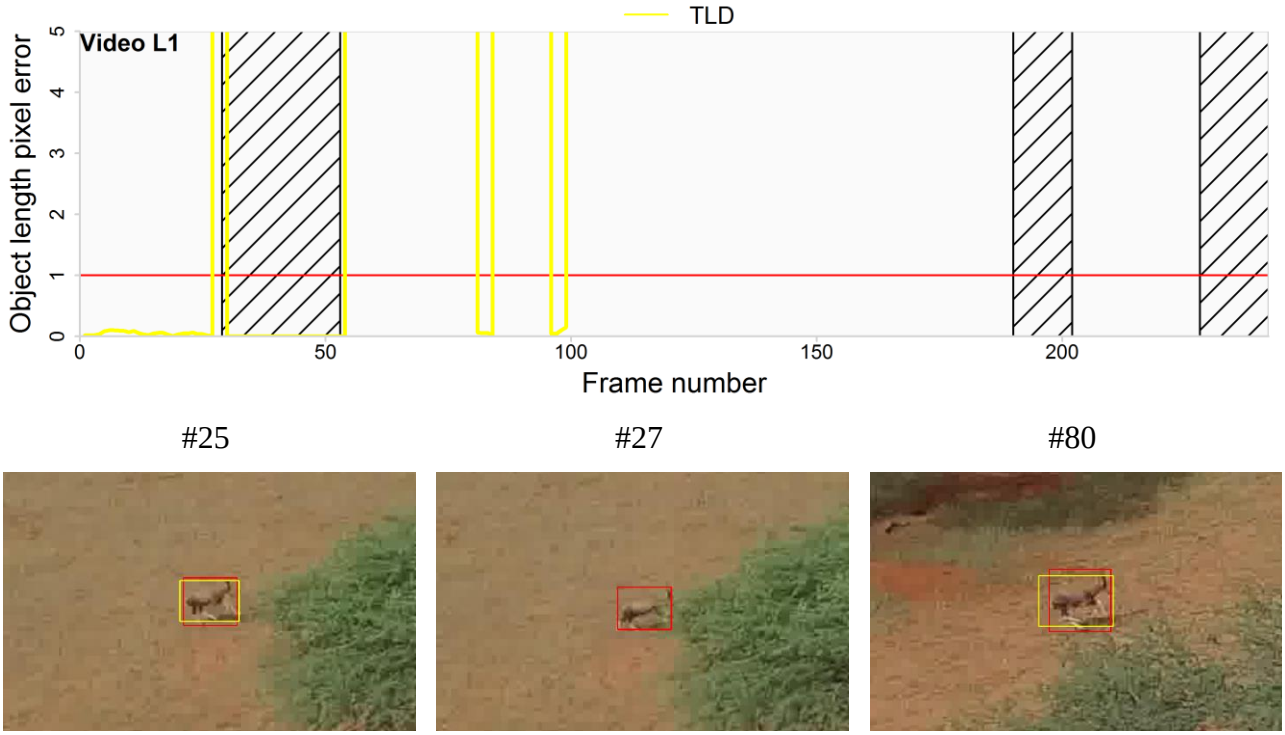


Figure 10: Performance plot of TLD on two occlusion events in video L1. The red line in the graph indicates the threshold of 1 object length and the black striped areas indicate two occlusion and one out-of-frame event. Given in the images are ground truth bounding boxes coloured in red and the predicted bounding boxes by TLD in yellow.

Re-detection

The TLD algorithm should be able to re-detect animals e.g. after occlusion events. Video L1 (Figure 10) already showed this capability, but only for a single animal. Video H2 (Figure 11) contained several small similar animals. In frame 100 the TLD still tracked the animal of interest correctly. However, from frame 129 till 201 it lost the main animal and shifted its focus to a similar object. In frame 202 it re-detected the animal of interest and tracked it for 94 frames after which the animal was lost again. Around frame 600 the tracker re-detected similar animals again, but not the animal of interest. The scale of the y-axis is increased in this graph to visualise re-detections of other animals.

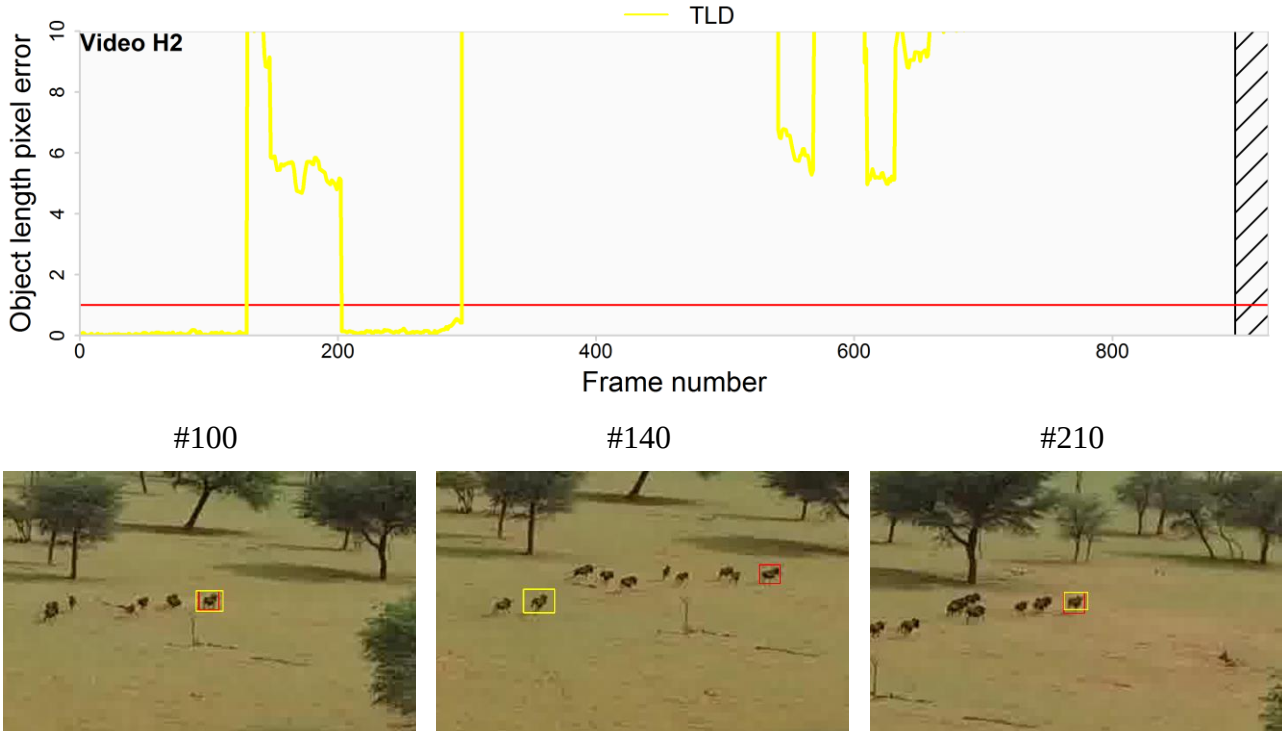


Figure 11: Performance plot of TLD on a re-detection event in video H2. The red line in the graph indicates the threshold of 1 object length and the black striped area indicates an out-of-frame event. Given in the images are ground truth bounding boxes coloured in red and the predicted bounding boxes by TLD in yellow. Keep in mind that the scale of the y-axis is increased in this graph.

7.3 Performance plots

Occlusion

Video C1 (Figure 12) shows the event of a long occlusion where the animal is lost for more than 40 frames. Additionally, the animal was relatively small and hardly visible due to the dark image. TLD was able to track the animal for a few frames, but only KLT was able to track the animal until it occluded behind a tree. However, it should be noted that the bounding box of KLT increased in size and thus did not completely overlap with the ground truth. After the occlusion, none of the trackers was able to re-detect the animal of interest.

The video O2 (Figure 13) shows a smaller occlusion event (13 frames). All trackers were able to track the animal for the first 100 frames. TLD and KLT, both trackers that succeeded until the occlusion event in the video C1 (Figure 12), failed after 160 frames. KCF, DCF and LSST tracked the animal until the short occlusion event (at frame 312), but they were unable to re-detect the animal after that event.

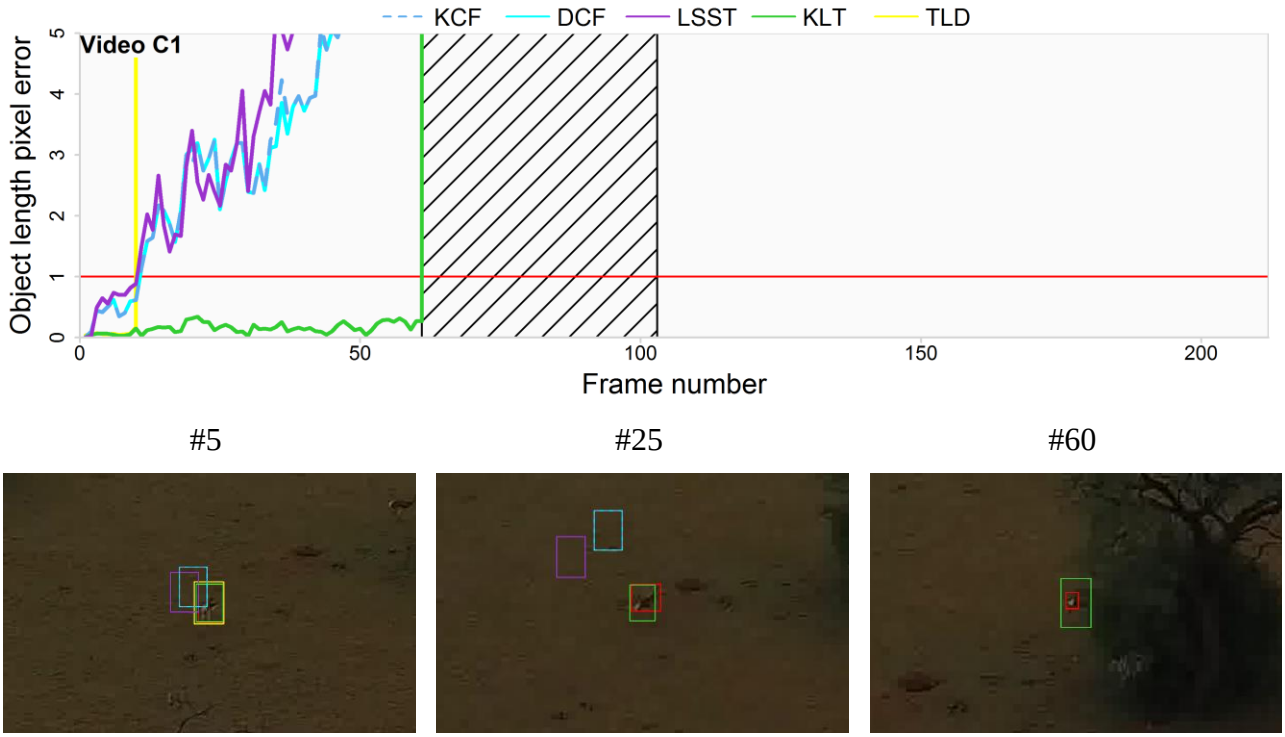


Figure 12: Performance plot of KCF (dotted blue), DCF (cyan), LSST (purple), KLT (green) and TLD (yellow) on one occlusion event in video C1. The red line in the graph indicates the threshold of 1 object length and the black striped areas indicate occlusion or out-of-frame events. The ground truth bounding box is coloured in red.

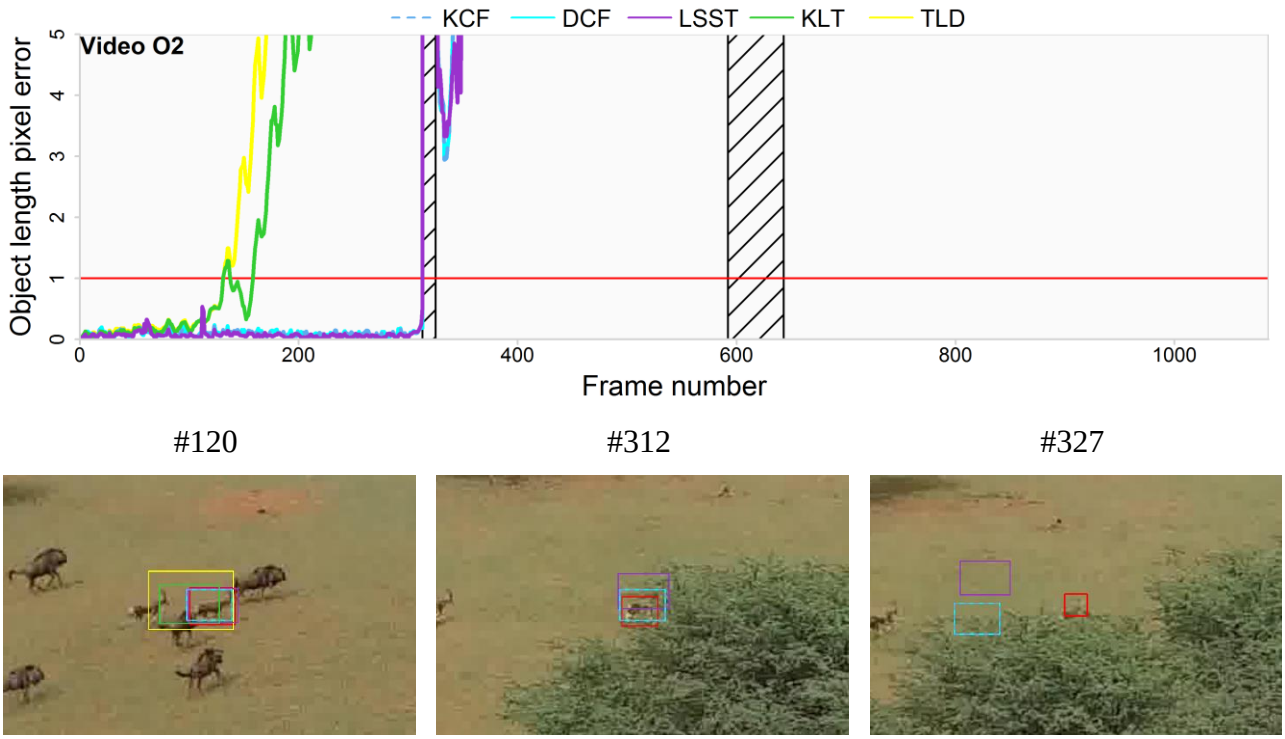


Figure 13: Performance plot of KCF (dotted blue), DCF (cyan), LSST (purple), KLT (green) and TLD (yellow) on two occlusion events in video O2. The red line in the graph indicates the threshold of 1 object length and the black striped areas indicate occlusion or out-of-frame events. The ground truth bounding box is coloured in red.

Similar objects

Video Q1 (Figure 14) contains a large herd of wildebeests as similar animals with object rotation and occlusion as additional characteristics. The animal of interest was clearly visible in the video with on average 3947 pixels in its bounding box. TLD and KLT lost the animal in an early stage, while LSST seemed to lose the animal only after it had entered the shade of a tree (frame 100). On the other hand, DCF and KCF were able to track the animal until it shortly occluded. After this, both trackers lost the animal but stayed in the vicinity. Frame 525 shows that this is not because of tracking a similar animal. In the end all trackers only tracked the animal of interest, without tracking any of the other wildebeests.

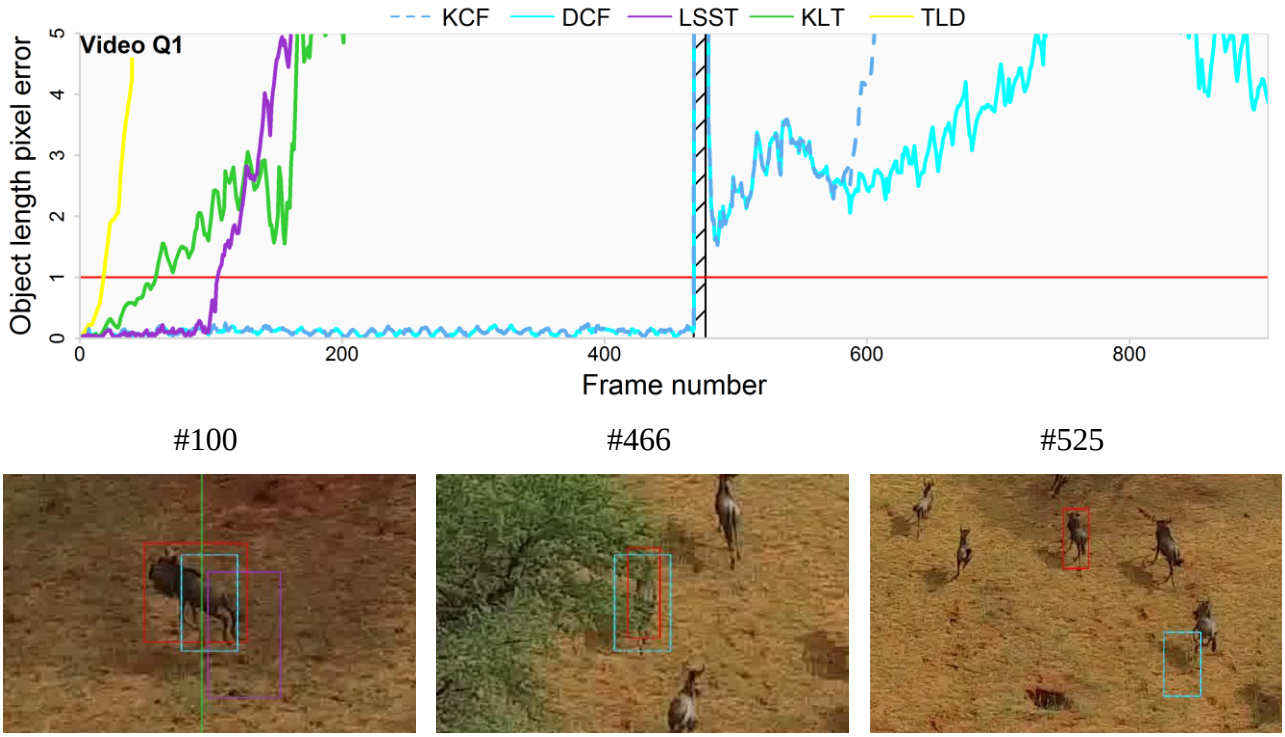


Figure 14: Performance plot of KCF (dotted blue), DCF (cyan), LSST (purple), KLT (green) and TLD (yellow) on video Q1 with the main characteristic of similar objects. Additionally the video contained object rotation and occlusion events. The red line in the graph indicates the threshold of 1 object length and the black striped area indicates and occlusion event. The ground truth bounding box is coloured in red.

Video D3 (Figure 15) contains five small (249-361 pixels), identical animals that run through the video from left to right. The trackers were assessed on all animals (video D1 till D5), but in video D3 the focus is on only one animal. LSST and TLD lost this animal after it faded into the dark background. TLD was able to re-detect it after the background became brighter again (frame 27). Around frame 55 all trackers seemed to gradually lose the animal. Here, the animal of interest caught up with one of the other animals, overlapped for a short moment and passed it. However, DCF, KCF and TLD all followed the other, slower moving animal. TLD was able to occasionally re-detect the animal of interest, as in frame 160. This image also shows that DCF and KCF still tracked the other animal. The distance between the two animals increased slowly, which explained the gradual increase in pixel error for both trackers.

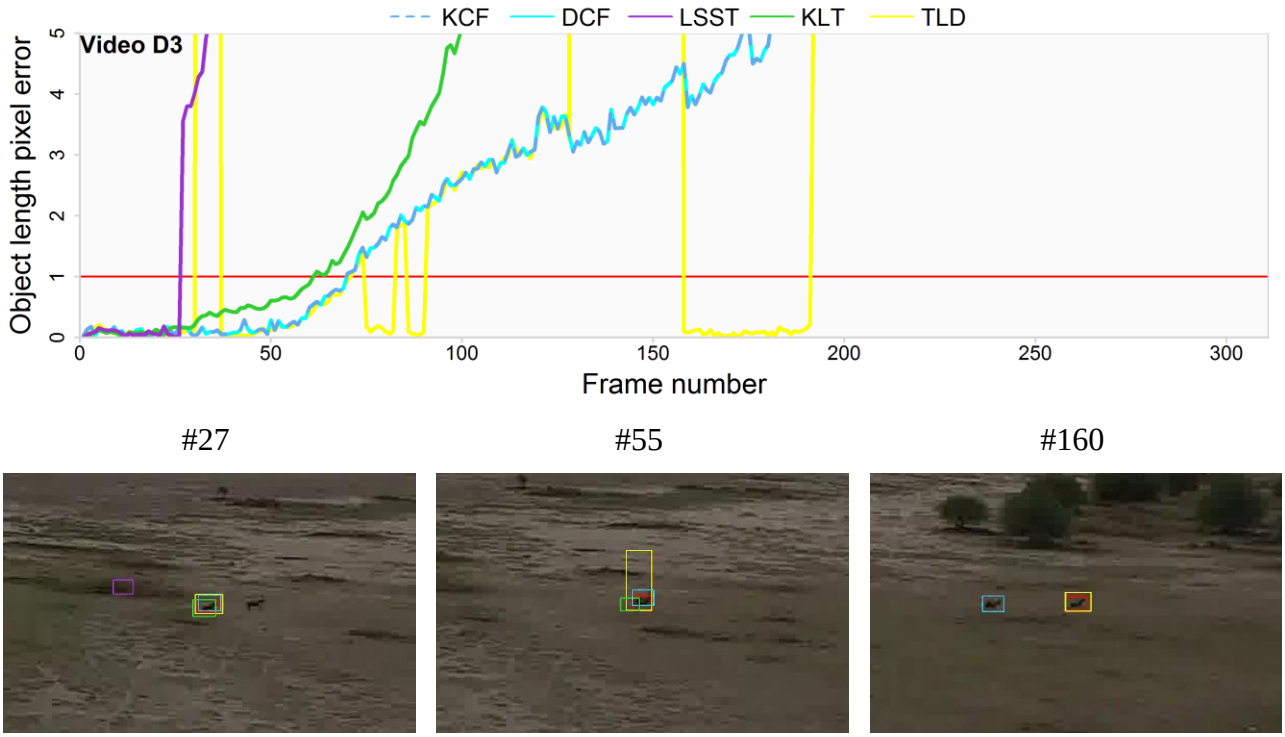


Figure 15: Performance plot of KCF (dotted blue), DCF (cyan), LSST (purple), KLT (green) and TLD (yellow) on video D3 with similar objects. The red line in the graph indicates the threshold of 1 object length. The ground truth bounding box is coloured in red.

Change of direction

The animals in video B2 and J1 are highlighted as particular cases, because they changed their direction in a sudden moment within a few frames (Figure 16 & 17). It is expected that the animals face a different side when they drastically change direction, which in turn influences the way a tracker recognises the object. In video B2 (Figure 16) the animal was recorded from a top-down perspective and ran to the left corner, while the UAV flew from bottom to top. KCF, DCF and LSST were not able to deal with the change of direction and lost the animal after a few frames already. KLT and TLD, on the other hand, tracked the animal until the last frame of the video.

The video of J1 (Figure 17) is a bit longer, compared to video B2. In the first 40 frames, the animal moved in a straight line towards the UAV position. Around frame 45 it turned around, and moved away from the UAV, which resulted in false detections by DCF and KCF. At frame 60 also TLD and LSST lost the animal of interest. The animal changed its representation, where it was now visible from the side. The only tracker that completed both videos in which the animal drastically changed direction, was KLT.

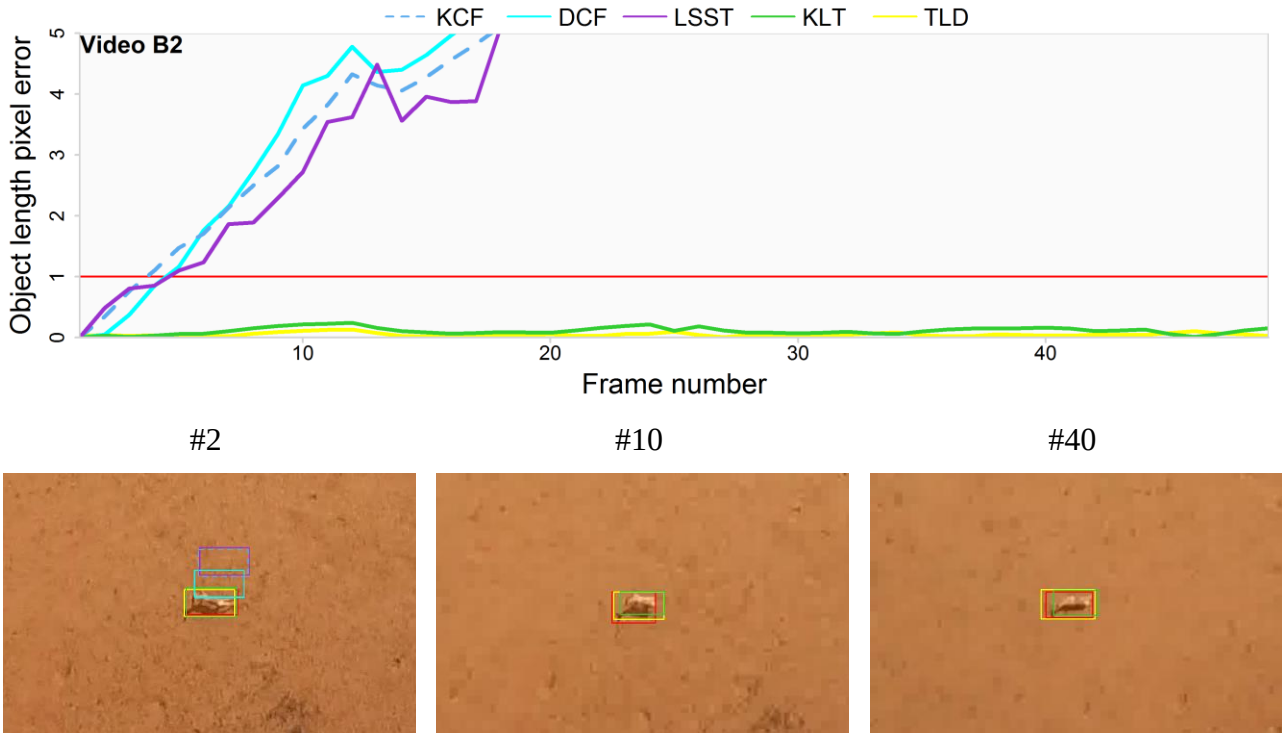


Figure 16: Performance plot of KCF (dotted blue), DCF (cyan), LSST (purple), KLT (green) and TLD (yellow) on video B2 with the object changing its direction. The red line in the graph indicates the threshold of 1 object length. Ground truth bounding boxes are coloured in red.

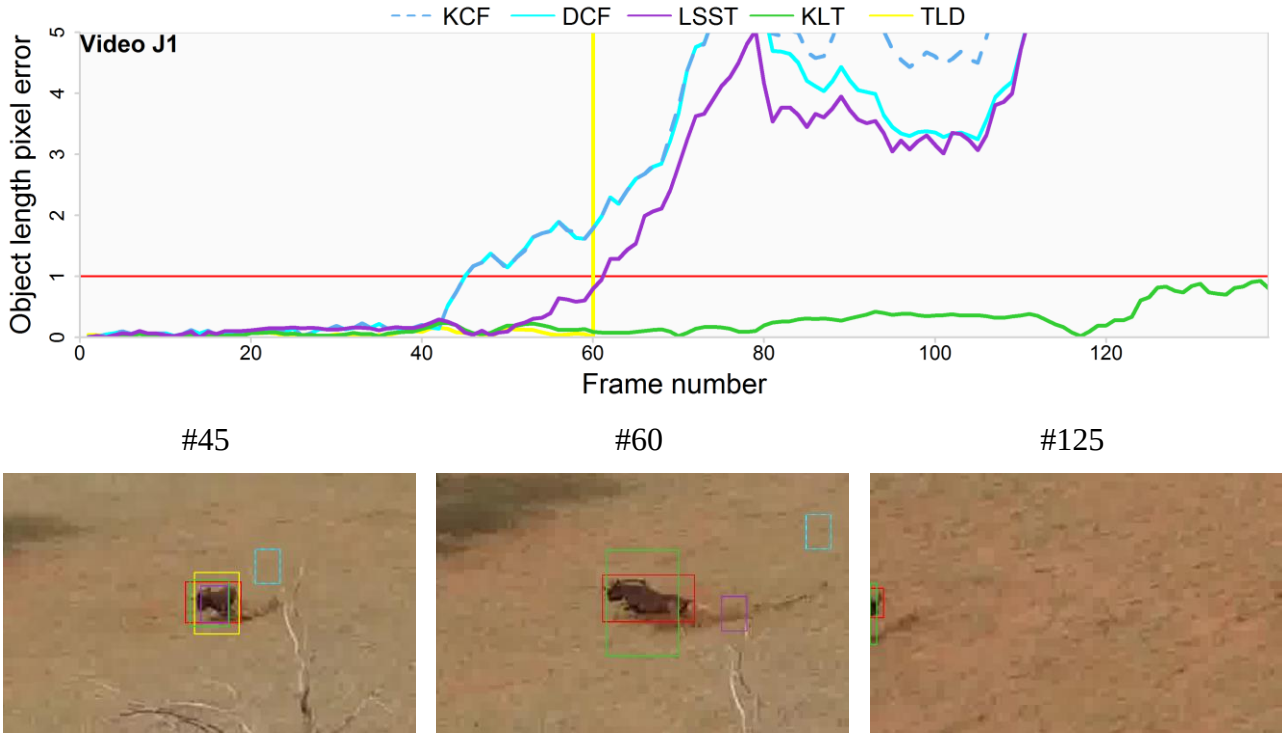


Figure 17: Performance plot of KCF (dotted blue), DCF (cyan), LSST (purple), KLT (green) and TLD (yellow) on video J1 with the object changing its direction. The red line in the graph indicates the threshold of 1 object length. The ground truth bounding box is coloured in red.

Camera movement

In the first 250 frames of video E3 (Figure 18) the UAV flew towards the animal, which resulted in magnified animal appearance. KCF, DCF and LSST were all able to deal with this animal enlargement. Around frame 280 it was noticeable that all trackers, except TLD, showed a steep increase in error distance, which was caused by rapid camera movement from left to right. No tracker was able to deal with the fast "animal movement", however it should be mentioned that they also did not lose the animal completely. This was a bit surprising, because after the camera movement, the animal of interest was not recorded from the side anymore, but more from behind and aside (see frame 400 and 700). This resulted in a complete different animal representation compared to the first 280 frames. After short occlusion events around frame numbers 590 and 800 the animal was re-detected by LSST and DCF. However, both bounding boxes did not overlap with the ground truth bounding box. The bounding box of LSST was not clearly visible in frame number 700, because it was only represented by a small dot.

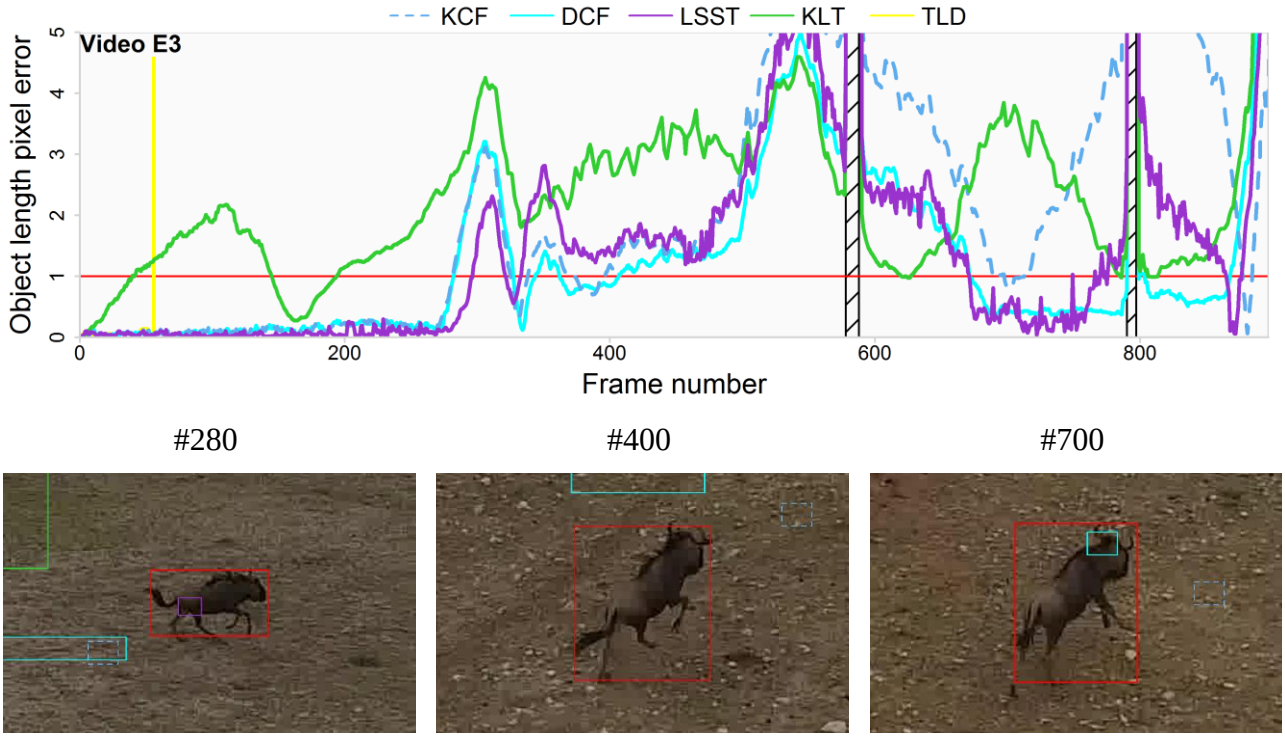


Figure 18: Performance plot of KCF (dotted blue), DCF (cyan), LSST (purple), KLT (green) and TLD (yellow) on video E3 with fast camera movement as main characteristic. The red line in the graph indicates the threshold of 1 object length and the black striped areas indicate occlusion events. The ground truth bounding box is coloured in red.

In video F1 the camera changed its direction immediately, which resulted in KCF, DCF and LSST losing the animal (Figure 19). After 49 frames also TLD lost the animal, where the animal was seen more from the side than in the initial frame. At last, the KLT tracker lost the animal in frame 290 where the animal was recorded from behind/aside. It should be mentioned that KCF, DCF and KLT stayed in the near vicinity of the animal, sometimes tracking predictions within the ground truth boundary.

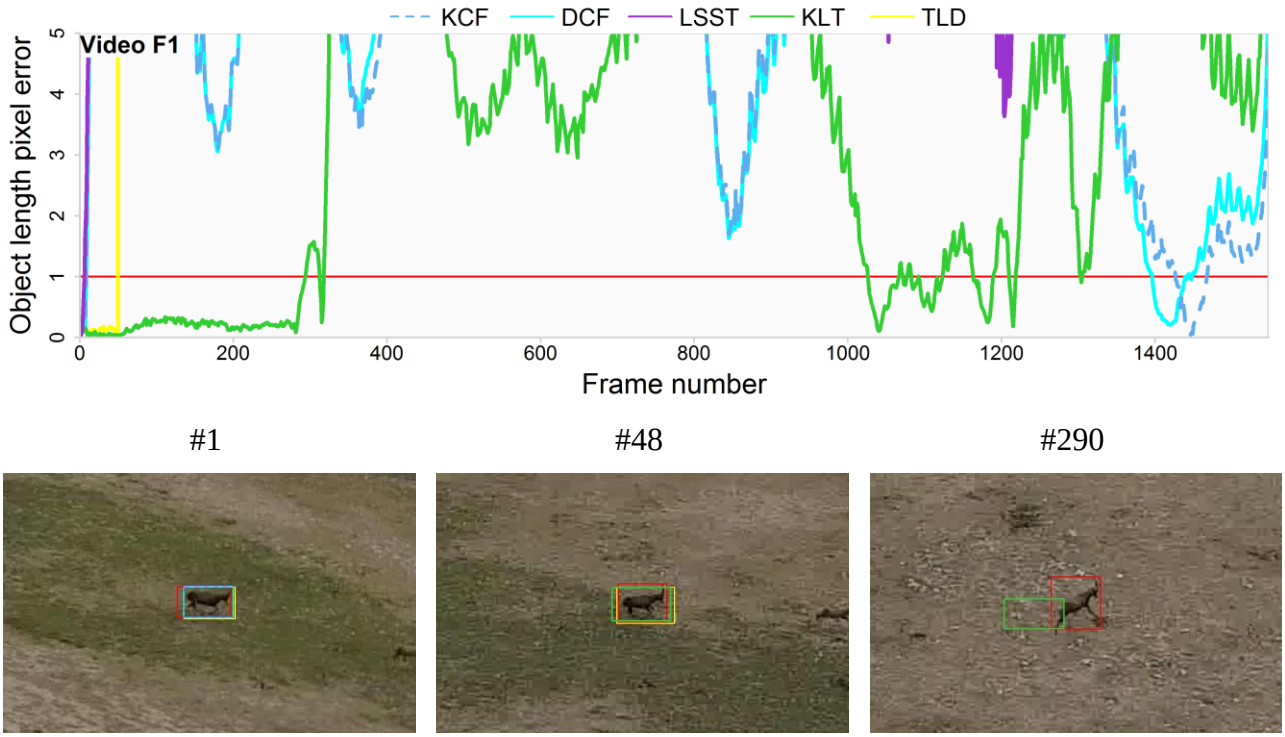


Figure 19: Performance plot of KCF (dotted blue), DCF (cyan), LSST (purple), KLT (green) and TLD (yellow) on video F1 with fast camera movement as the main characteristic. The red line in the graph indicates the threshold of 1 object length. The ground truth bounding box is coloured in red.

Object rotation

In video G2 (Figure 20) a group of five gemsboks are fleeing from the approaching UAV. As with the objects in videos D1-D5 (Figure 15 for object D3), video G2 only focused on one of the five animals as well. They started from standing position and within 25 frames they completely changed their appearance. This caused TLD, KLT and LSST to lose track of them. KCF and DCF were able to follow the animal of interest, despite that it was changing its appearance all the time to lose the UAV. However, in frame 430 the camera moved from left to right, while the animal of interest moved in the opposite direction. This is shown in frame 430 and 460, which are at the original dimensions of the image to show the quick movement. In 30 frames, less than a second, the object moved 1105 pixels to the left. From this point on, no successful tracking records had been made by any of the trackers.

Video K1 (Figure 21) consists of several identical animals that are changing their appearance gradually throughout the video. It was expected that this gradual change would be picked up particularly well by the TLD tracker, which is designed to update its feature representation to account for such a gradual change. However, in the first frame TLD already lost the animal. LSST and KLT also lost the animal as soon as it started to change its appearance. Only KCF and DCF were able to track the complete trajectory of the animal successfully. Here, it becomes visible that those two trackers did not update their bounding box, which caused the mismatch in overlap with the ground truth bounding box.

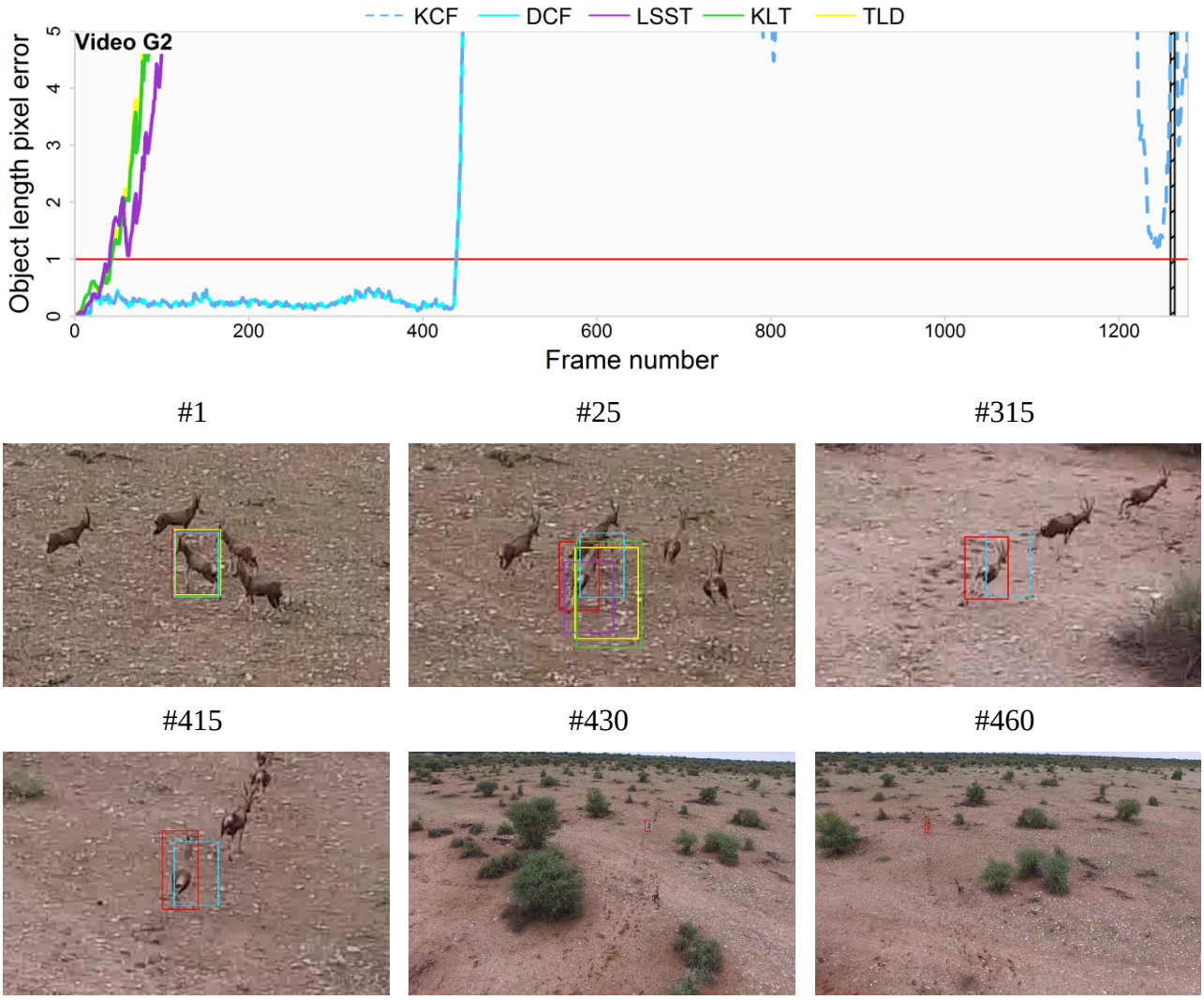


Figure 20: Performance plot of KCF (dotted blue), DCF (cyan), LSST (purple), KLT (green) and TLD (yellow) on video G2 with the object changing its direction. Additional characteristics are camera movement (from left to right starting at frame 430) and similar objects. The red line in the graph indicates the threshold of 1 object length. The ground truth bounding box is coloured in red.

In the last video focusing on object rotation, most trackers were able to follow the animal for the first 40 frames (Figure 22). After the animal changed its appearance, LSST and TLD lost it. KCF, DCF and KLT were able to track the animal until frame 90, where it changed its appearance even more and increased in scale. In the end, KLT seemed to track the animal correctly, because the centre of its predicted bounding box was within the limits of the ground truth bounding box. However, the surface of the bounding box did not overlap anymore.

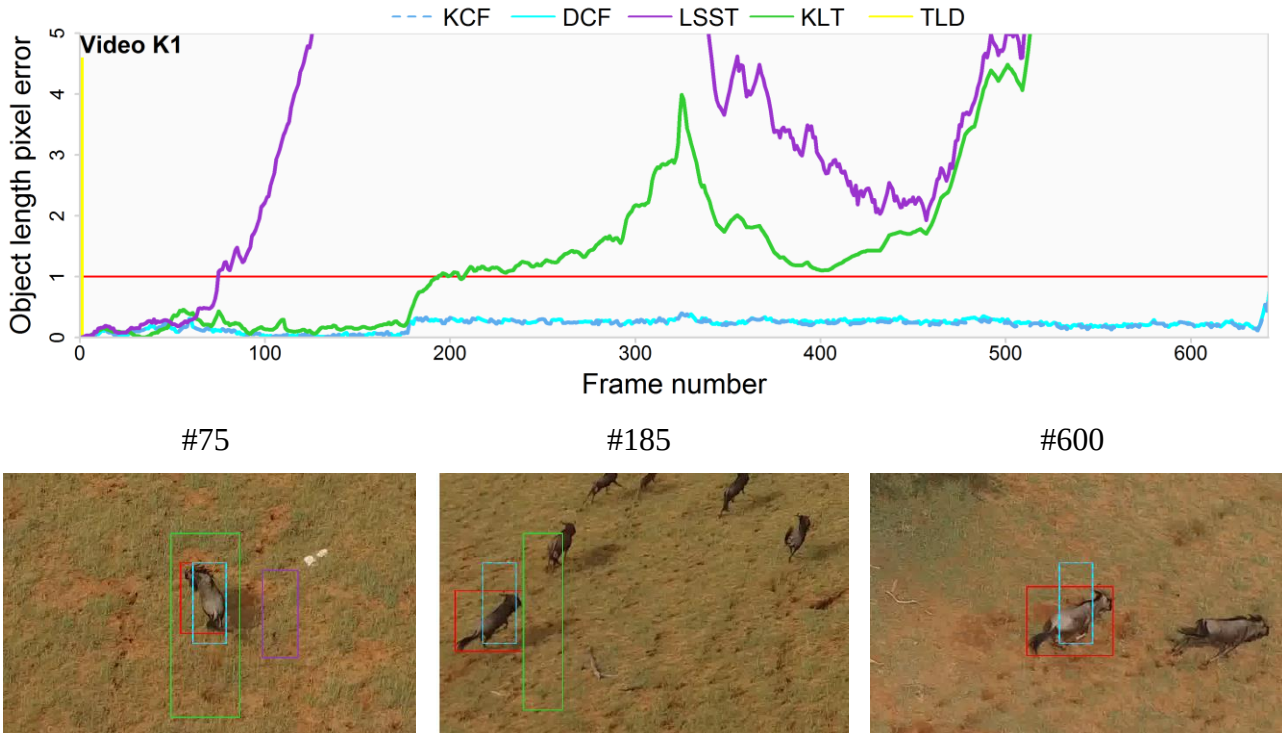


Figure 21: Performance plot of KCF (dotted blue), DCF (cyan), LSST (purple), KLT (green) and TLD (yellow) on video K1 with the object changing its direction and similar objects. The red line in the graph indicates the threshold of 1 object length. The ground truth bounding box is coloured in red.

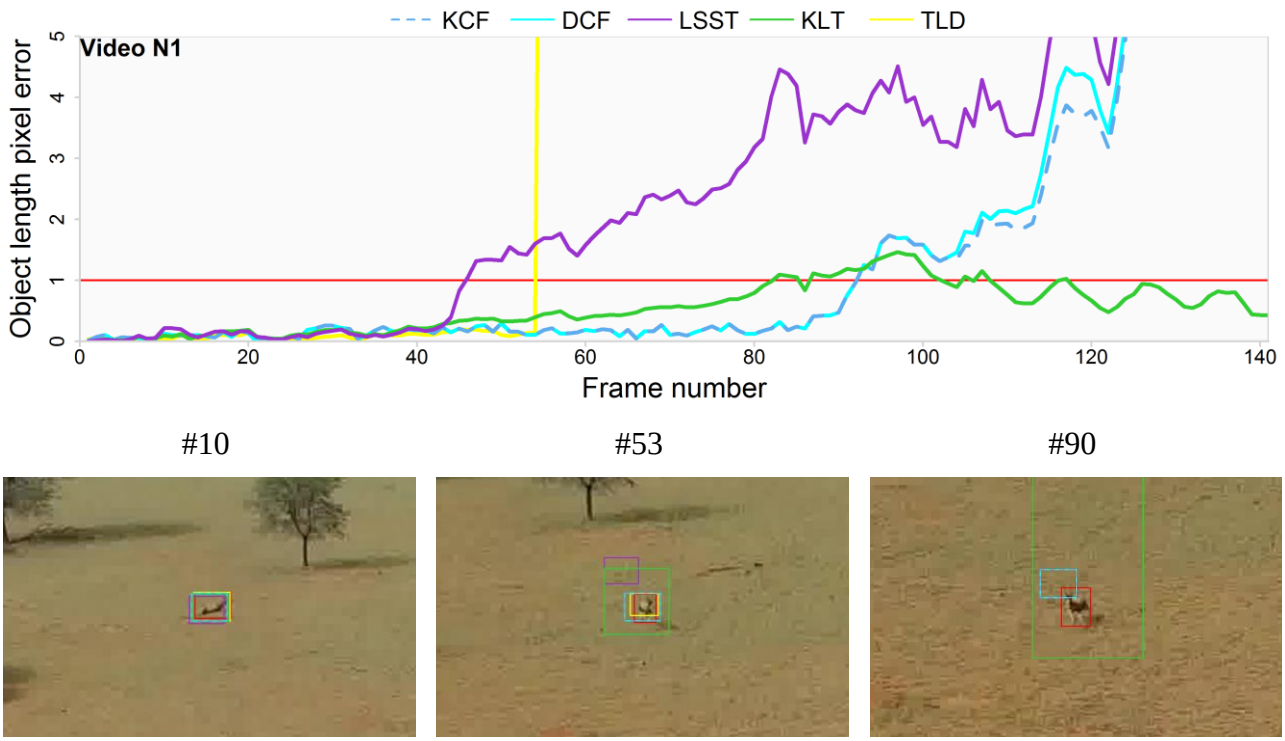


Figure 22: Performance plot of KCF (dotted blue), DCF (cyan), LSST (purple), KLT (green) and TLD (yellow) on video N1 with the object changing its direction. The red line in the graph indicates the threshold of 1 object length. The ground truth bounding box is coloured in red.

7.4 Performance Detector-Tracking system

The results in section 7.1 and 7.3 shows that the KCF on HOG features outperforms the other trackers on the KWT dataset. Therefore, this tracker is used in the detector-tracking system. Beforehand, it was expected that the additional detector would reinforce the performance of KCF. However, it turned out that the new developed system does not outperform KCF on HOG features. The corresponding AUC values, KCF (14.64) and the new system (10.95), show that there is a decrease of almost 25%. This means that KCF on HOG outperformed the new system on several videos (Figure 23).

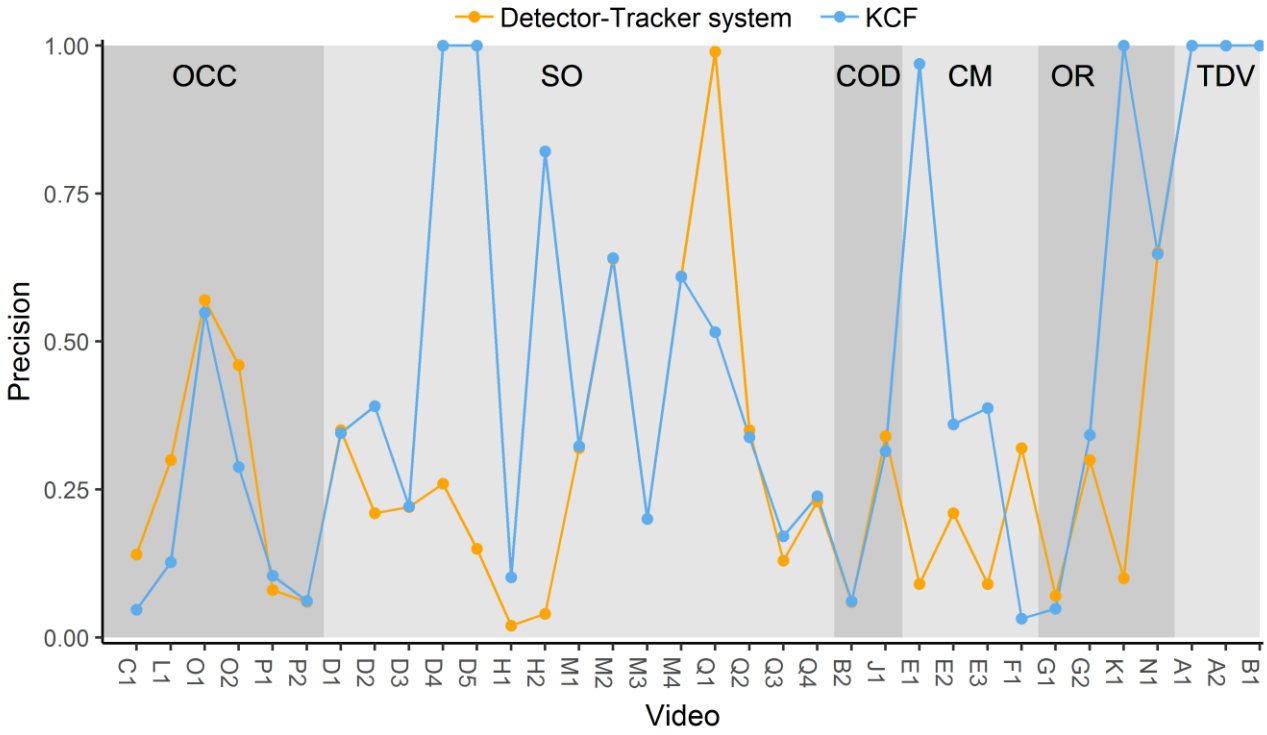


Figure 23: Precision results for comparing KCF on HOG features and the detector-tracking system for all 34 videos in the KWT dataset. The videos are ordered according to their main characteristic: occlusion (OCC), similar objects (SO), change object direction (COD), camera movement (CM), object rotation (OR) or top-down view (TDV). The precision is calculated by dividing the number of true positives by the total number of frames with occlusion, but without out-of-frame events.

In 19 of the 34 videos, the new detector-tracking system performed almost similarly as the KCF tracker (Figure 23 or Table 2). In the remaining 15 videos either KCF or the new system showed better performances compared to the other. It was found that KCF outperformed the new system especially in videos with similar objects, such as in video H2 (Figure 24). After 30 frames, the new system lost the animal of interest and tracked a similar object instead. It is likely that the detector could not find the object of interest at all, since it re-initialised on a different object at an early stage of the video. In the process of re-initialisation, the closest detection was found on the object in the orange bounding box at frame 30. Therefore, the new system decided to track this object instead. It also happened that the new system lost the object completely, as visualised by frame 400. The short re-detection at frame 810 can be seen as a coincidence, where the animal accidently passed the 'searching' bounding box.

Besides similar animals, the new system could also be lost on other objects as in video D4 (Figure 25). At frame 89 the new system lost the animal of interest out of nothing. It tracked a darker patch for at least 50 frames, instead of the nearby moving object. Other characteristics where the new system failed were camera movement (video E1-E3) and object rotation (video K1).

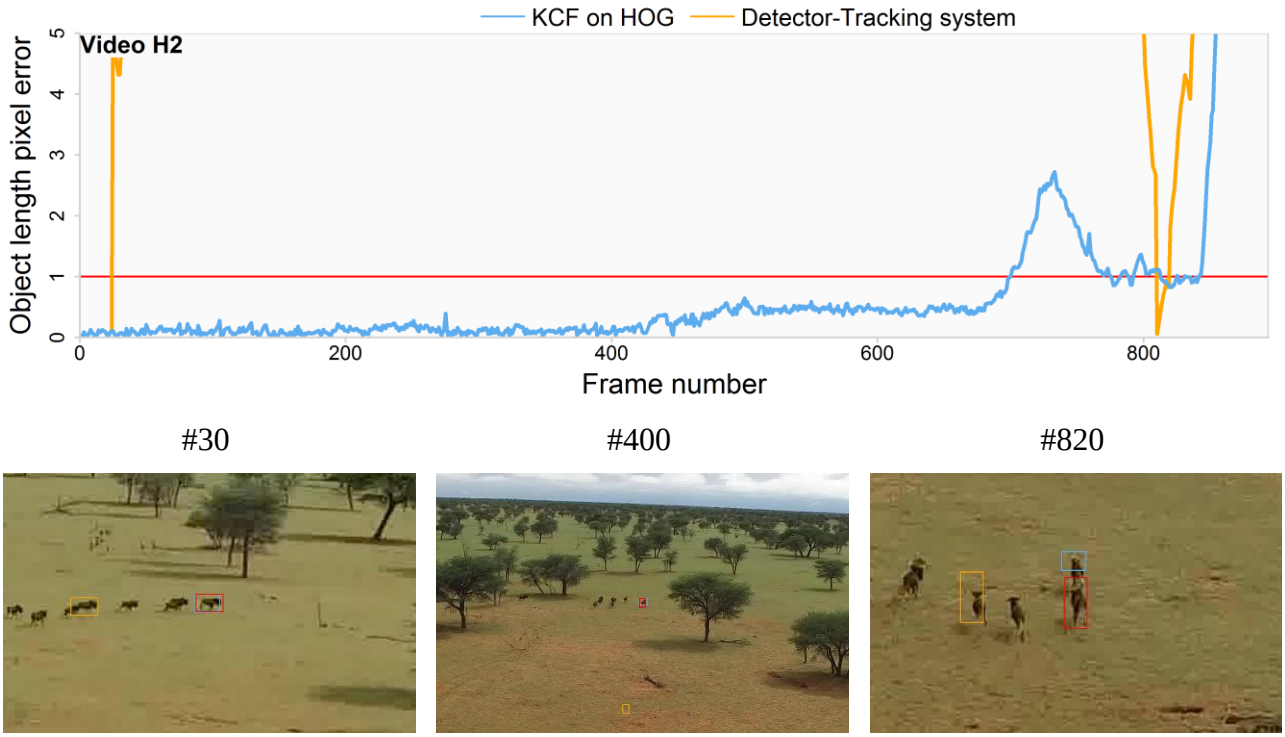


Figure 24: Performance plot of KCF (blue) and the new proposed detector-tracking system (orange) on video H2. The red line in the graph indicates the threshold of 1 object length. The ground truth bounding box is coloured in red.

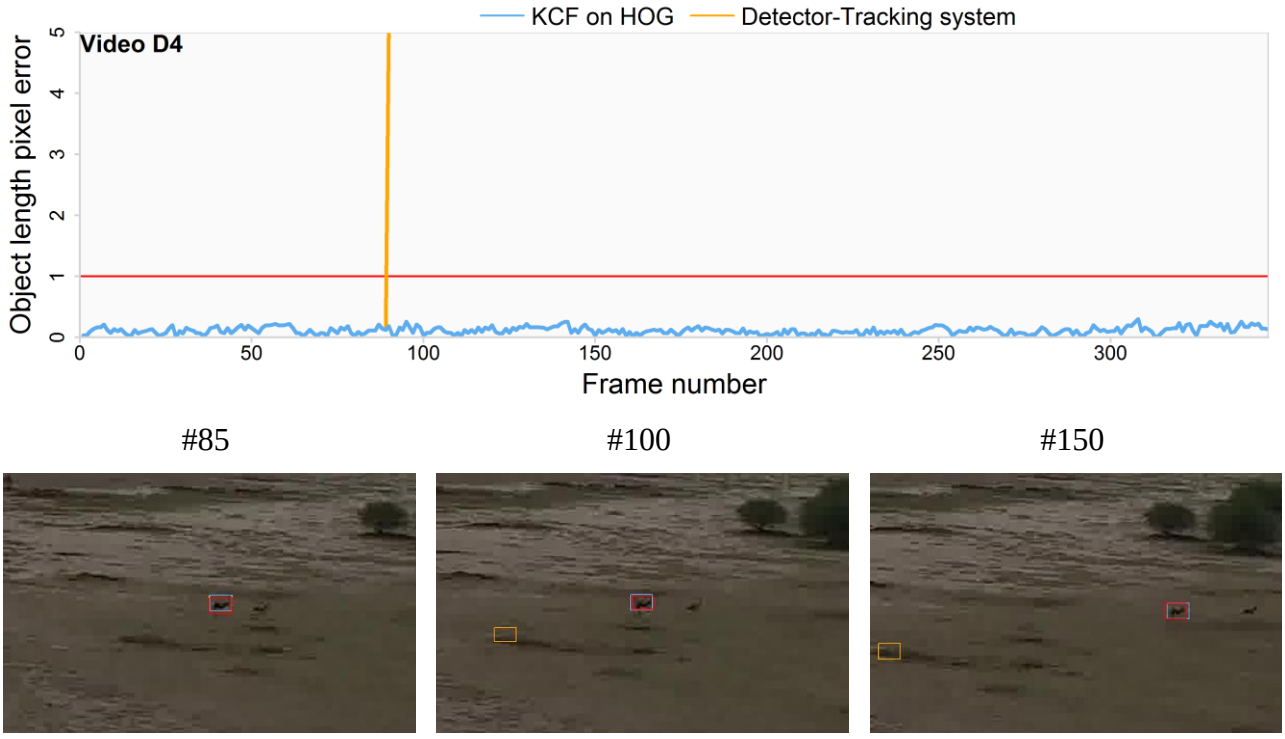


Figure 25: Performance plot of KCF (blue) and the new proposed detector-tracking system (orange) on video D4. The red line in the graph indicates the threshold of 1 object length. The ground truth bounding box is coloured in red.

In 5 videos the new system outcompeted KCF. Interesting to see was that the new system could handle occlusion events now (Figure 26). In this video the KCF lost the animal of interest after the occlusion event, while the new system was able to track it and fulfil tracking until the end of the video. A similar result was found in video O2, where the animal of interest was also re-detected after a short occlusion event. However, after a longer occlusion event in the same video, the new system lost the object.

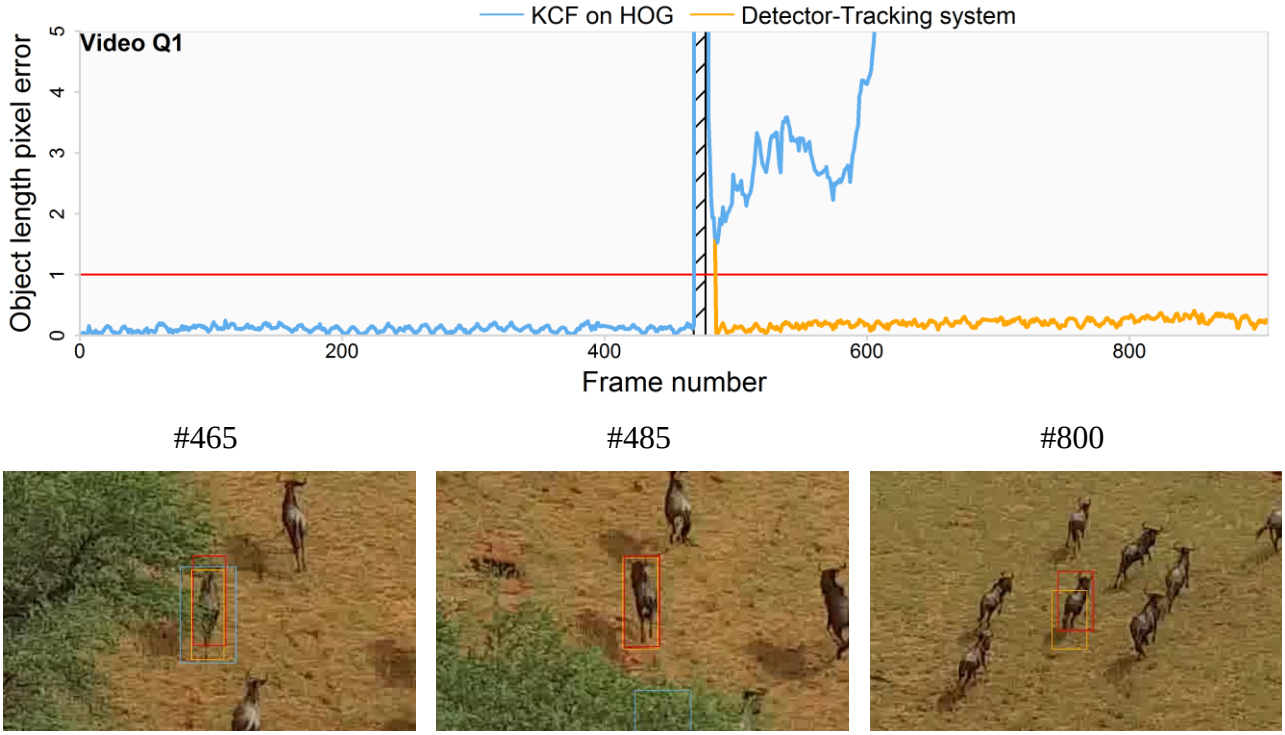


Figure 26: Performance plot of KCF (blue) and the new proposed detector-tracking system (orange) on video D4. The graph visualises the improvement of the new system on occlusion events. The red line in the graph indicates the threshold of 1 object length. The ground truth bounding box is coloured in red.

7.5 Summary of the results

KCF showed the best results on the overall performance, where AUC values indicated that it marginally outcompeted DCF. Both trackers performed relatively well on similar objects (Q1 in Figure 13) and objects changing their appearance (G2, K1, N1 in Figure 19, 20 and 21 respectively or Table 2). However, they failed in occasions where the object changed its direction or occlusion events. TLD was expected to perform better in those events and it was indeed able to handle out-of-frame (Figure 8) and occlusion events (Figure 9). However, in most cases it failed to re-detect the objects. The performance of KLT on some occasions was surprising: for example, in video C1 (Figure 11), it was the only tracker able to track the dark, small object. Also, it performed well in videos where objects suddenly changed their direction (B2, J1 in Figure 15 and 16 respectively). LSST performed in most cases comparable to KCF and DCF, except when objects changed appearance (Q1, G2, K1, N1 in Figure 13, 19, 20 and 21 respectively) or with small similar objects (D3 in Figure 14), where it lost the objects of interest.

The new system performed similar as KCF in 19 of the 34 videos. In videos with similar object, the new system failed more easily compared to KCF (Figure 23). However, it was shown that the new system can handle occlusion events (Figure 26).

Table 2: Overview with the precision values per tracker on characteristic events of the KWT dataset. The orange underlined values indicate that a tracker made it at least until the occlusion event in that video. The green underlined values indicate that a tracker fulfilled the complete video.

		Detector-Tracking system	KCF	DCF	LSST	KLT	TLD
Occlusion	C1	0.14	0.05	0.05	0.05	0.29	0.05
	L1	0.30	0.13	0.13	0.13	0.13	0.26
	O1	0.57	0.55	0.55	0.14	0.14	0.12
	O2	0.46	0.29	0.29	0.29	0.14	0.12
	P1	0.08	0.10	0.10	0.03	0.01	0.01
	P2	0.06	0.06	0.06	0.05	0.07	0.04
Similar objects	D1	0.35	0.35	0.35	0.02	0.05	0.19
	D2	0.21	0.39	0.39	0.10	0.18	0.20
	D3	0.22	0.22	0.22	0.08	0.20	0.36
	D4	0.25	1.00	1.00	1.00	0.78	0.18
	D5	0.15	1.00	1.00	0.15	0.73	0.57
	H1	0.02	0.10	0.10	0.25	0.08	0.05
	H2	0.04	0.82	0.82	0.80	0.31	0.25
	M1	0.32	0.32	0.33	0.19	1.00	0.92
	M2	0.64	0.64	0.64	0.10	0.76	0.46
	M3	0.20	0.20	0.33	0.18	0.96	0.93
	M4	0.61	0.61	0.61	0.20	0.98	0.98
	Q1	0.99	0.52	0.52	0.11	0.06	0.02
	Q2	0.35	0.34	0.34	0.09	0.10	0.03
	Q3	0.13	0.17	0.17	0.16	0.11	0.00
	Q4	0.23	0.24	0.24	0.25	0.16	0.00
Change Object Direction	B2	0.06	0.06	0.08	0.08	1.00	1.00
	J1	0.34	0.31	0.31	0.44	1.00	0.43
Camera Movement	E1	0.09	0.97	0.97	0.97	0.14	0.21
	E2	0.21	0.36	0.35	0.36	0.11	0.02
	E3	0.09	0.39	0.59	0.46	0.11	0.06
	F1	0.32	0.03	0.04	0.00	0.27	0.03
Object Rotation	G1	0.07	0.05	0.05	0.08	0.04	0.03
	G2	0.30	0.34	0.34	0.03	0.03	0.03
	K1	0.10	1.00	1.00	0.12	0.30	0.00
	N1	0.65	0.65	0.65	0.32	0.85	0.38
Top-Down View	A1	1.00	1.00	1.00	1.00	1.00	1.00
	A2	1.00	1.00	1.00	0.98	1.00	0.41
	B1	1.00	1.00	1.00	0.06	1.00	1.00

8 Discussion

8.1 Evaluation on the tracker's performances

KCF and DCF showed the best performances on the KWT dataset, where KCF slightly outperformed DCF. Both trackers showed good performances on videos with similar objects and objects changing their appearance (Table 2). Other than in video D3, where the object of interest was occluded by a similar object, no failures were detected in videos with similar objects as long as they did not come too close into the near vicinity. The videos of G2, K1 and N1 with changing object appearance are also correctly tracked to a certain extent. This ability to track objects changing their appearance is allocated to discriminative trackers by Smeulders et al. (2014), but Bibi and Ghanem (2015) assign this ability to generative trackers. It is expected that indeed discriminative trackers are in favour of dealing with appearance changes, since they mainly focus on the classification of object or background (Asha & Narasimhadhan, 2017). They do not build a complex model of the object, like generative trackers, which does not match the candidate patches anymore after the object changed its appearance. Smeulders et al. (2014) also state that this type of trackers are dealing better with changing object size compared to generative trackers. This is due to the sampling strategy, which takes both the object and its surrounding background into account. KCF and DCF make use of a cyclic shift to sample positive and negative patches around the object of interest. The positive patches are translations of the object to deal with appearance change and the negative patches contain background samples to exclude against (Bibi & Ghanem, 2015; Smeulders et al., 2014). In combination with updating the base patches, this makes it possible to deal with events such as background clutter or changes in appearance or scale (Henriques et al., 2015; Ma et al., 2015). Both trackers failed on occlusion events, which is not surprising since there is no failure recovery mechanism implemented (Henriques et al., 2015). Events in which the object abruptly changes its direction and camera motion are also prone to failures. The failures on camera motion appear in contrast with findings of Smeulders et al. (2014), who found that trackers based on Gaussian-based motion models perform relatively well under camera ego-motion. The fast change in appearance probably caused failure in updating the classifier and thus tracking the object. In the end it can be stated that both trackers performed well according to their capabilities.

The relatively simplest tracker, KLT, performed well on some specific events. Smeulders et al. (2014) already found that simple trackers are not inferior towards advanced trackers. KLT was the only tracker capable of tracking the animal in video C1 and it achieved good performance on objects drastically changing their direction (Table 2). It kept tracking the objects as long as there were small feature windows that did not change too much compared to their initial state. For videos B2 and J1, that contained objects drastically changing their directions, KLT was able to save some feature windows to correctly predict the trajectory of the object. However, it should be mentioned that those videos only took 81 and 140 frames (2 and 4 seconds respectively). Since the tracker did not update the feature windows, nor create new windows during the process, it was considered unsuitable for long tracking performances. This could also explain why the tracker failed in events like object rotation. Such events often cause the tracker to gradually lose feature windows, which in the end results in completely losing the object. KLT is able to track an object as long as it does not lose too many feature windows.

Beforehand, it was expected that TLD would perform well on events like fast moving objects, changing object size and occlusion (Smeulders et al., 2014; D. Wang et al., 2013; Wu et al., 2013). It achieved outstanding results on camera motion events in the assessment of Smeulders et al. (2014) and it had the ability to re-detect the object after occlusion events which should allow the tracker to perform well on long videos. However, it failed on almost all events in the KWT dataset and surprisingly it was often not able to re-detect the object of interest (Table 2). It did notice when an object was out-of-frame or occluded, but it did not re-detect the object after such an event. In some cases TLD was able to occasionally re-detect an object, but only for short periods or it re-detected other, similar objects. The failure in re-detection probably originates from the detection part of the tracker. Since similar

objects are probably going through the first and second phase of the tracker's pipeline, it is expected that it fails in the third phase where a nearest neighbour classifier is applied. Especially if the object changed its appearance after occlusion, the detector in TLD could allocate the tracker to the wrong object or considers it as lost. This is also described by Kalal et al. (2012) as one of the limitations of TLD. They state that in those cases the object can only be re-detected if it takes one of its positions that was seen before by the tracker. In case TLD makes a false positive or if it has an imprecise object representation, the learning component may cause drift. This updating process amplifies errors in cases of wrong object representations (Santner et al., 2010; Wu et al., 2013). The algorithm learns the wrong object appearance and thus detects and tracks the wrong object. It should be noted that only the detector component is updated with new object representations, causing the tracker to make the same mistakes over and over again (Kalal et al., 2012). In other studies, it was found that TLD often fails in case of illumination or change in scale and appearance (Smeulders et al., 2014; D. Wang et al., 2013; Wu et al., 2013). Especially the results of changes in scale and appearance were found in the KWT dataset as well. TLD seemed to lose objects changing their appearance or scale more often compared to KCF and DCF. All three trackers make use of negative samples to distinct the object from its background. According to Henriques et al. (2015) TLD samples around hundreds of images around the object, compared to thousands of images used by KCF and DCF. This makes the classifier of TLD less strong compared to the one of KCF and DCF, which results in a lower performance score for TLD on videos where the object changes its scale or appearance. In the end, TLD did not perform as well as was expected. However, a re-detection component after occlusion would improve many trackers and is almost essential in wildlife tracking datasets like KWT.

8.2 Evaluation on the detector-tracker system

The detector-tracking system performed worse compared to KCF on HOG features. It failed especially when similar objects were present. As described in section 6.2, the tracker only starts the re-initialisation process if the tracker's bounding box did not match with a detection for five consecutive frames. This means that the object detector was not able to find the object of interest. It made a lot of false positives, without a true positive. This is likely to be the result of the training process. The object detector was trained on a different dataset, with only videos that recorded the object from a top-down perspective. In contrast, the KWT dataset contained mostly objects recorded from oblique perspective. This would explain why the detector could not find the object of interest in some situations and thus started the re-initialisation process. Based on the animal's speed and direction, several potential locations are determined. If an object of similar appearance shows up near those locations and it is detected by the detector, then the new system assumes this is the object of interest. Objects similar in appearance could be animals of the same species, but also a similar looking rock. This failure would be less likely to happen if the detector makes more true positives, because in that case the re-initialisation process is not started.

The new detector-tracking system showed that it can handle short occlusion events. Due to the linear interpolation, the system is able to re-detect the object as soon as it is visible again. However, longer occlusion events are still prone to failure. The longer an occlusion event takes, the higher the chance that the object deviates from its linear line and the more noise is added to 'correct' observations. Those correct positions are acquired from the 5, 10 and 15th last frame. They are not correct anymore if an occlusion lasts for more than 20 frames. In literature, occlusion events are often considered as short or partial occlusions. However, in wildlife tracking it is likely that occlusion events can take longer than 15 frames.

8.3 HOG features or raw image pixels

It was found in literature that both KCF and DCF performed better on HOG features than on raw image pixels (Bibi & Ghanem, 2015; Henriques et al., 2015). HOG features have a stronger representational

power compared to raw pixels (N. Wang et al., 2015) and therefore, a fair comparison between KCF and DCF on HOG features and the other trackers on raw pixels was actually not possible. On the other hand, by using HOG features for KCF and DCF, they are assessed in the way they are presented in the original papers. It also fits the objective of this research to combine the most suitable tracker with a detector to develop an optimal wildlife monitoring system. Therefore, KCF on HOG features was used in the joint detector-tracking system.

Additionally, the KCF was run on raw image pixels to see if HOG features are indeed favoured over raw image pixels. It was found that KCF performed better on raw image pixels than on HOG features, which is in contrast with the findings of Henriques et al. (2015) and Bibi and Ghanem (2015). It should be mentioned that this extra analysis was performed after developing the detector-tracking system and that no further additions could be made anymore. It is stated by Ma et al. (2015) that HOG features are less effective in discriminating the object from a cluttered background (i.e. object has the same colour as the background). Several videos in the KWT dataset have the trouble of background clutter, for example video A2, B2, C1 or D1-D5. Henriques et al. (2015) used the dataset of Wu et al. (2013)[4] to test their KCF on both raw pixels and HOG features. This dataset also contained videos with the characteristic background clutter (32 of 98 videos). This means that in both studies the characteristic background clutter had its influence on the performance of KCF on HOG features. Further research should focus on the question why KCF on raw image pixels outperformed KCF on HOG features for the KWT dataset? Which other characteristics have an influence on those results? And which feature is most suitable for wildlife tracking? If this follow-up research shows that indeed raw pixels are favoured for wildlife tracking, then the detector-tracking system can be easily adapted to this.

8.4 Quality of KWT dataset

One of the contributions of this study was a video dataset with annotated wildlife objects. The dataset contains high resolution images that are comparable to wildlife videos used for monitoring purposes. The dataset seems to be quite challenging if the achieved results are compared with other studies. TLD achieved an AUC value of 4.93 in this study, where in the study of Wu et al. (2013) this value was between 50 and 60. On the same dataset, KCF on HOG performed really well with AUC values ranging from 66 until 75 (Henriques et al., 2015). The low AUC values in this study might have to do with the small objects that are tracked and the challenging events like occlusion, object rotation, similar objects, camera motion and abrupt changes of direction. The KWT dataset also contains videos for long-term and multi-object tracking. There are 7 occasions in total (videos F1, G1, G2, O1, O2, P1 and P2), over 4 videos, in which the object was present for more than 1000 frames (with a maximum of 1575 frames). For comparison, in the WAT dataset of Risse et al. (2017) the maximum number of frames was 710. The KWT dataset also allows for multiple object tracking, since some videos contain large herds of animals. It should be noted that not every individual has been annotated, since herds could be formed by up to 30 animals. The CAVIAR[5] dataset is well known for multiple object tracking. However, it contains multiple videos, mostly from surveillance cameras, with humans as object. According to my knowledge, the KWT dataset is the first dataset that allows for long-term and multiple object tracking with videos of wild African animals recorded from UAV perspective.

[4] Available on: http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html
[5] Available on: http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/

9 Conclusion

This research assessed the performance of five unsupervised trackers on the KWT dataset, with the objective to develop a detector-tracking system. The KWT dataset contained five main characteristics: occlusion, similar objects, change of object direction, camera movement and object rotation. It was found that KCF on HOG features performed best on the entire dataset. It achieved slightly better results than DCF on HOG features, but outcompeted the other trackers on raw pixels. Both KCF and DCF showed the best performance on characteristic events like similar objects and changing appearance (object rotation). They failed on events like abrupt change of direction and occlusion. LSST showed in general comparable results as KCF and DCF, except for objects changing their appearance. In those kind of occasions LSST easily lost the object. KLT performed surprisingly well on events where objects drastically changed their direction. However, it failed on events like camera motion, objects changing appearance and occlusion. At last, TLD failed in most of the events that the trackers had to deal with. It was able to recognize out-of-frame and occlusion events (i.e. it noticed when an object was not visible), but it hardly re-detected objects after occlusion. In some cases, it tracked a similar object. In the end, it can be concluded that KCF on HOG features performed best on the KWT dataset. Therefore, this tracker was implemented with an object detector with the idea that this detector could re-initialise the tracker in case it lost the object. However, the developed detector-tracking system performed worse than KCF on HOG features, especially on videos with similar objects. On the other hand, the new system was able to handle short occlusion events.

10 Recommendations

10.1 UAVs in wildlife monitoring

UAV based video imagery has the potential to be an optimal data source for tracking wildlife and hence it is already applied in several monitoring programmes (Bevan et al., 2015; Ditmer et al., 2015; Gonzalez et al., 2016; Vermeulen et al., 2013). However, at this moment the system is not perfect and there are some limitations. Among these, the disturbance caused by the aircraft on animals is noteworthy. The disturbance is related to the altitude on which the UAV operates and there needs to be a trade-off between the quality of the object in the video and the amount of disturbance. The trackers perform better on large objects that are clearly visible, but this results in unnatural behaviour of the animals. At this moment, the videos in the dataset consist of UHD resolution (3840 x 2160 pixels), which is used for most videos nowadays. The smallest objects of 249 pixels (±16 x 16 pixels) can even be seen from this resolution, as proved by the KWT dataset. It is expected that even smaller objects can be tracked in the near future due to improvements in UAV and video technology. UAVs are expected to become quieter and thus can fly lower (Chabot, 2018), while video resolution is likely to increase even more.

Another aspect to consider when applying UAVs to wildlife monitoring is the flight perspective. In the KWT dataset only 4 videos were taken from top-down perspective, while the remaining 14 videos were taken from oblique perspective. Both perspectives have their strengths and weaknesses. Videos recorded from the oblique perspective contain mostly large, clearly visible objects and videos are not always bound to be taken by UAVs, but can also be taken from a car or tower. However, the KWT dataset shows that the animals from this perspective are obviously disturbed by the UAV. The animals can also take different appearances due to rotation in the 3D environment, with the result that the tracker has to train its classifier on more object representations. On the other hand, animals from the 2D top-down perspective are only recorded from above and thus have less difference in representations. Thereby, objects cannot be occluded behind other species anymore, it is easier for the pilot to anticipate to abrupt changes in direction and small objects are still detectable from higher altitudes as shown in the KWT dataset. Considering the advantages and weaknesses of both perspectives and the expected developments in UAV and image technology, UAV recordings from top-down perspective are likely to be more useful for animal tracking compared than from oblique perspective.

10.2 Steps-to-take

This study forms a basis for further research on an automated wildlife monitoring system using UAV video data. It aimed at providing starting points by showing the performance of several unsupervised trackers. By looking specifically at the performance of the trackers on certain events and linking those results to the specific components of the tracker, it is possible to combine valuable components with each other into one tracker that can deal with several characteristics. In this way a new tracker could be developed on the components that contributed to the good performance.

Another possibility would be to improve the developed detector-tracking system, for example by making it aware of out-of-frame events. This would require some adaptations in the last step of the re-initialisation process. At this moment, the detector-tracking system checks whether the distance between a detection and a potential coordinate is within the object's boundary box. If this is the case, then the tracker is re-initialised with the coordinates proposed by the detector. However, nothing happens when this is not the case. It is likely that with an out-of-frame event this re-initialisation process did not happen for several times and after a certain time it is expected that the object is out-of-frame. This could be combined with the position in the image (e.g. close to the edges), so that long occlusion events are not accidentally discarded.

Another improvement could be to use a second order polynomial function, instead of a linear function, to predict the potential locations. This would increase the chance of a correct prediction in situations where the animal bends its direction a bit. It is expected that overfitting of the polynomial function will not happen, since the parameters are derived from only three points. However, follow-up research should indicate whether a second order polynomial function is indeed an added value compared to the linear function.

It would also be worthwhile to test the performance of the new system on HOG features or raw image pixels. A small internal survey indicated that KCF on raw pixels might perform better than KCF on HOG features. It would be worthwhile to investigate this more elaborately and to focus on the question why raw pixels might work better than HOG features. At last, the detector could be trained on a more similar dataset with also object from an oblique perspective to acquire more true positive detections. In the end, all those steps should contribute to a more accurate and robust wildlife monitoring system.

Literature

Aarts, G., MacKenzie, M., McConnell, B., Fedak, M., & Matthiopoulos, J. (2008). Estimating space-use and habitat preference from wildlife telemetry data. *Ecography, 31*(1), 140-160.

Acevedo, J., Aguayo-Lobo, A., Allen, J., Botero-Acosta, N., Capella, J., Castro, C., . . . Flórez-González, L. (2017). Migratory preferences of humpback whales between feeding and breeding grounds in the eastern South Pacific. *Marine Mammal Science, 33*(4), 1035-1052.

Aebischer, N. J., Robertson, P. A., & Kenward, R. E. (1993). Compositional analysis of habitat use from animal radio-tracking data. *Ecology, 74*(5), 1313-1325.

Andrew, W., Greatwood, C., & Burghardt, T. (2017). *Visual Localisation and Individual Identification of Holstein Friesian Cattle via Deep Learning.* Paper presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

Asha, C., & Narasimhadhan, A. (2017). Robust infrared target tracking using discriminative and generative approaches. *Infrared Physics & Technology, 85*, 114-127.

Barbu, T. (2014). Pedestrian detection and tracking using temporal differencing and HOG features. *Computers & Electrical Engineering, 40*(4), 1072-1079.

Benhamou, S., Valeix, M., Chamaillé-Jammes, S., Macdonald, D. W., & Loveridge, A. J. (2014). Movement-based analysis of interactions in African lions. *Animal Behaviour, 90*, 171-180.

Bevan, E., Wibbels, T., Najera, B. M., Martinez, M. A., Martinez, L. A., Martinez, F. I., . . . Hernandez, M. H. (2015). Unmanned aerial vehicles (UAVs) for monitoring sea turtles in near-shore waters. *Marine Turtle Newsletter*(145), 19.

Beyer, H. L., Haydon, D. T., Morales, J. M., Frair, J. L., Hebblewhite, M., Mitchell, M., & Matthiopoulos, J. (2010). The interpretation of habitat preference metrics under use–availability designs. *Philosophical Transactions of the Royal Society B: Biological Sciences, 365*(1550), 2245-2254.

Bibi, A., & Ghanem, B. (2015). *Multi-template scale-adaptive kernelized correlation filters.* Paper presented at the Proceedings of the IEEE International Conference on Computer Vision Workshops.

Boitani, L., & Fuller, T. (2000). *Research techniques in animal ecology: controversies and consequences*: Columbia university press.

Borji, A., Cheng, M.-M., Jiang, H., & Li, J. (2015). Salient object detection: A benchmark. *IEEE Transactions on Image Processing, 24*(12), 5706-5722.

Bouguet, J.-Y. (2001). Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corporation, 5*(1-10), 4.

Branson, K., Robie, A. A., Bender, J., Perona, P., & Dickinson, M. H. (2009). High-throughput ethomics in large groups of Drosophila. *Nature methods, 6*(6), 451.

Bridge, E. S., Thorup, K., Bowlin, M. S., Chilson, P. B., Diehl, R. H., Fléron, R. W., . . . Robinson, W. D. (2011). Technology on the move: recent and forthcoming innovations for tracking migratory birds. *BioScience, 61*(9), 689-698.

Brudvig, L. A., Damschen, E. I., Haddad, N. M., Levey, D. J., & Tewksbury, J. J. (2015). The influence of habitat fragmentation on multiple plant–animal interactions and plant reproduction. *Ecology, 96*(10), 2669-2678.

Buchheit, M., Allen, A., Poon, T. K., Modonutti, M., Gregson, W., & Di Salvo, V. (2014). Integrating different tracking systems in football: multiple camera semi-automatic system, local position measurement and GPS technologies. *Journal of sports sciences, 32*(20), 1844-1857.

Burghardt, T., & Ćalić, J. (2006). Analysing animal behaviour in wildlife videos using face detection and tracking. *IEE Proceedings-Vision, Image and Signal Processing, 153*(3), 305-312.

Čehovin, L., Leonardis, A., & Kristan, M. (2016). Visual object tracking performance measures revisited. *IEEE Transactions on Image Processing, 25*(3), 1261-1274.

Chabot, D. (2018). Trends in drone research and applications as the Journal of Unmanned Vehicle Systems turns five. *Journal of Unmanned Vehicle Systems, 6*(1), vi-xv.

Chabot, D., & Bird, D. M. (2012). Evaluation of an off-the-shelf unmanned aircraft system for surveying flocks of geese. *Waterbirds, 35*(1), 170-174.

Cooke, S. J., Nguyen, V. M., Murchie, K. J., Thiem, J. D., Donaldson, M. R., Hinch, S. G., . . . Fisk, A. (2013). To tag or not to tag: Animal welfare, conservation, and stakeholder considerations in fish tracking studies that use electronic tags. *Journal of International Wildlife Law & Policy, 16*(4), 352-374.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning, 20*(3), 273-297.

Dalal, N., & Triggs, B. (2005). *Histograms of oriented gradients for human detection.* Paper presented at the Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on.

Danelljan, M., Hager, G., Shahbaz Khan, F., & Felsberg, M. (2015). *Learning spatially regularized correlation filters for visual tracking.* Paper presented at the Proceedings of the IEEE International Conference on Computer Vision.

Dell, A. I., Bender, J. A., Branson, K., Couzin, I. D., de Polavieja, G. G., Noldus, L. P., . . . Wikelski, M. (2014). Automated image-based tracking and its application in ecology. *Trends in ecology & evolution, 29*(7), 417-428.

Dewhirst, O. P., Evans, H. K., Roskilly, K., Harvey, R. J., Hubel, T. Y., & Wilson, A. M. (2016). Improving the accuracy of estimates of animal path and travel distance using GPS drift-corrected dead reckoning. *Ecology and Evolution, 6*(17), 6210-6222.

Dey, C. J., & Quinn, J. S. (2014). Individual attributes and self-organizational processes affect dominance network structure in pukeko. *Behavioral Ecology, 25*(6), 1402-1408.

Ditmer, M. A., Vincent, J. B., Werden, L. K., Tanner, J. C., Laske, T. G., Iaizzo, P. A., . . . Fieberg, J. R. (2015). Bears show a physiological but limited behavioral response to unmanned aerial vehicles. *Current Biology, 25*(17), 2278-2283.

Ellenberg, U., Mattern, T., & Seddon, P. J. (2013). Heart rate responses provide an objective evaluation of human disturbance stimuli in breeding birds. *Conservation physiology, 1*(1).

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision, 88*(2), 303-338.

Geismann, P., & Schneider, G. (2008). *A two-staged approach to vision-based pedestrian recognition using Haar and HOG features.* Paper presented at the Intelligent Vehicles Symposium, 2008 IEEE.

Gill, J. A., Alves, J. A., Sutherland, W. J., Appleton, G. F., Potts, P. M., & Gunnarsson, T. G. (2014). Why is timing of bird migration advancing when individuals are not? *Proceedings of the Royal Society of London B: Biological Sciences, 281*(1774), 20132161.

Gonzalez, L. F., Montes, G. A., Puig, E., Johnson, S., Mengersen, K., & Gaston, K. J. (2016). Unmanned Aerial Vehicles (UAVs) and artificial intelligence revolutionizing wildlife monitoring and conservation. *Sensors, 16*(1), 97.

Grémillet, D., Puech, W., Garçon, V., Boulinier, T., & Le Maho, Y. (2012). Robots in ecology: welcome to the machine. *Open Journal of Ecology, 2*(2), 49-57.

Hare, S., Golodetz, S., Saffari, A., Vineet, V., Cheng, M.-M., Hicks, S. L., & Torr, P. H. (2016). Struck: Structured output tracking with kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 38*(10), 2096-2109.

Henriques, J. F., Caseiro, R., Martins, P., & Batista, J. (2015). High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 37*(3), 583-596.

Hodgson, A., Kelly, N., & Peel, D. (2013). Unmanned aerial vehicles (UAVs) for surveying marine fauna: a dugong case study. *PloS one, 8*(11), e79556.

Hong, S., You, T., Kwak, S., & Han, B. (2015). *Online tracking by learning discriminative saliency map with convolutional neural network.* Paper presented at the International Conference on Machine Learning.

Huber, P. J. (1964). Robust estimation of a location parameter. *The annals of mathematical statistics*, 73-101.

Hunter, A. (2007). *Sensor-based animal tracking*.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning* (Vol. 112): Springer.

Ji, S. (2016). Kernel method. Retrieved from https://commons.wikimedia.org/wiki/User:Shiyu_Ji

Joshi, K. A., & Thakore, D. G. (2012). A survey on moving object detection and tracking in video surveillance system. *International Journal of Soft Computing and Engineering, 2*(3), 44-48.

Kalal, Z., Matas, J., & Mikolajczyk, K. (2009). *Online learning of robust object detectors during unstable tracking.* Paper presented at the Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on.

Kalal, Z., Mikolajczyk, K., & Matas, J. (2010). *Forward-backward error: Automatic detection of tracking failures.* Paper presented at the Pattern recognition (ICPR), 2010 20th international conference on.

Kalal, Z., Mikolajczyk, K., & Matas, J. (2012). Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 34*(7), 1409-1422.

Kays, R., Crofoot, M. C., Jetz, W., & Wikelski, M. (2015). Terrestrial animal tracking as an eye on life and planet. *Science, 348*(6240), aaa2478.

Kellenberger, B., Volpi, M., & Tuia, D. (2017). Fast Animal Detection in UAV Images Using Convolutional Neural Networks. *MultiModal Remote Sensing, University of Zurich*.

Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Čehovin, L., . . . Chi, Z. (2016) The visual object tracking VOT2016 challenge results. In*: Vol. 9914 LNCS. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (pp. 777-823).

Kristan, M., Matas, J., Leonardis, A., Felsberg, M., Cehovin, L., Fernández, G., . . . Pflugfelder, R. (2015). *The visual object tracking vot2015 challenge results.* Paper presented at the Proceedings of the IEEE international conference on computer vision workshops.

Kristan, M., Pflugfelder, R., Leonardis, A., Matas, J., Porikli, F., Cehovin, L., . . . Gatt, A. (2013). *The visual object tracking vot2013 challenge results.* Paper presented at the Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). *Imagenet classification with deep convolutional neural networks.* Paper presented at the Advances in neural information processing systems.

LASIG. (2016, 24-10-2017). SAVMAP. Retrieved from http://lasig.epfl.ch/savmap

Li, Z., Ding, B., Wu, F., Lei, T. K. H., Kays, R., & Crofoot, M. C. (2013). Attraction and avoidance detection from movements. *Proceedings of the VLDB Endowment, 7*(3), 157-168.

Long, J. A., Nelson, T. A., Webb, S. L., & Gee, K. L. (2014). A critical examination of indices of dynamic interaction for wildlife telemetry studies. *Journal of Animal Ecology, 83*(5), 1216-1233.

Lucieer, A., Turner, D., King, D. H., & Robinson, S. A. (2014). Using an Unmanned Aerial Vehicle (UAV) to capture micro-topography of Antarctic moss beds. *International Journal of Applied Earth Observation and Geoinformation, 27*, 53-62.

Ma, C., Yang, X., Zhang, C., & Yang, M.-H. (2015). *Long-term correlation tracking.* Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.

Mallick, S. (2016). Histogram of Oriented Gradients. Retrieved from https://www.learnopencv.com/histogram-of-oriented-gradients/

Manohar, N., Kumar, Y. S., & Kumar, G. H. (2018). An Approach for the Development of Animal Tracking System. *International Journal of Computer Vision and Image Processing (IJCVIP), 8*(1), 15-31.

Millspaugh, J. J., & Marzluff, J. M. (2001). Radio-tracking and animal populations: past trends and future needs. In *Radio tracking and animal populations* (pp. 383-393): Elsevier.

Mulero-Pázmány, M., Stolper, R., Van Essen, L., Negro, J. J., & Sassen, T. (2014). Remotely piloted aircraft systems as a rhinoceros anti-poaching tool in Africa. *PloS one, 9*(1), e83873.

Ning, J., Zhang, L., Zhang, D., & Wu, C. (2010). Interactive image segmentation by maximal similarity based region merging. *Pattern Recognition, 43*(2), 445-456.

Ojha, S., & Sakhare, S. (2015). *Image processing techniques for object tracking in video surveillance-A survey.* Paper presented at the Pervasive Computing (ICPC), 2015 International Conference on.

Rey, N., Volpi, M., Joost, S., & Tuia, D. (2017). Detecting animals in African Savanna with UAVs and the crowds. *Remote Sensing of Environment, 200*, 341-351.

Risse, B., Mangan, M., Del Pero, L., & Webb, B. (2017). *Visual Tracking of Small Animals in Cluttered Natural Environments Using a Freely Moving Camera.* Paper presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

Rowcliffe, J. M., Jansen, P. A., Kays, R., Kranstauber, B., & Carbone, C. (2016). Wildlife speed cameras: measuring animal travel speed and day range using camera traps. *Remote Sensing in Ecology and Conservation, 2*(2), 84-94.

Rutz, C., & Hays, G. C. (2009). New frontiers in biologging science. In: The Royal Society.

Santner, J., Leistner, C., Saffari, A., Pock, T., & Bischof, H. (2010). *PROST: Parallel robust online simple tracking.* Paper presented at the Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on.

Sbalzarini, I. F., & Koumoutsakos, P. (2005). Feature point tracking and trajectory analysis for video imaging in cell biology. *Journal of structural biology, 151*(2), 182-195.

Shi, J. (1994). *Good features to track.* Paper presented at the Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on.

Smeulders, A. W., Chu, D. M., Cucchiara, R., Calderara, S., Dehghan, A., & Shah, M. (2014). Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 36*(7), 1442-1468.

Spampinato, C., Chen-Burger, Y.-H., Nadarajan, G., & Fisher, R. B. (2008). Detecting, Tracking and Counting Fish in Low Quality Unconstrained Underwater Videos. *VISAPP (2), 2008*(514-519), 1.

Spiegel, O., Leu, S. T., Bull, C. M., & Sih, A. (2017). What's your move? Movement as a link between personality and spatial dynamics in animal populations. *Ecology letters, 20*(1), 3-18.

Stanley, C. Q., McKinnon, E. A., Fraser, K. C., Macpherson, M. P., Casbourn, G., Friesen, L., . . . Diggs, N. E. (2015). Connectivity of wood thrush breeding, wintering, and migration sites based on range-wide tracking. *Conservation Biology, 29*(1), 164-174.

Tang, F., Brennan, S., Zhao, Q., & Tao, H. (2007). *Co-tracking using semi-supervised support vector machines.* Paper presented at the Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on.

Tomasi, C., & Kanade, T. (1991). Detection and tracking of point features.

van Gemert, J. C., Verschoor, C. R., Mettes, P., Epema, K., Koh, L. P., & Wich, S. (2014). *Nature Conservation Drones for Automatic Localization and Counting of Animals.* Paper presented at the ECCV Workshops (1).

Vermeulen, C., Lejeune, P., Lisein, J., Sawadogo, P., & Bouché, P. (2013). Unmanned aerial survey of elephants. *PloS one, 8*(2), e54700.

Viola, P., & Jones, M. (2001). *Rapid object detection using a boosted cascade of simple features.* Paper presented at the Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on.

Viola, P., & Jones, M. J. (2004). Robust real-time face detection. *International Journal of Computer Vision, 57*(2), 137-154.

Vondrick, C., Patterson, D., & Ramanan, D. (2013). Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision, 101*(1), 184-204.

Wang, D., Lu, H., & Yang, M.-H. (2013). *Least soft-threshold squares tracking.* Paper presented at the Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on.

Wang, N., Shi, J., Yeung, D.-Y., & Jia, J. (2015). *Understanding and diagnosing visual tracking systems.* Paper presented at the Computer Vision (ICCV), 2015 IEEE International Conference on.

Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., & Vapnik, V. (2001). *Feature selection for SVMs.* Paper presented at the Advances in neural information processing systems.

White, G. C., & Garrott, R. A. (2012). *Analysis of wildlife radio-tracking data*: Elsevier.

Wikelski, M., Kays, R. W., Kasdin, N. J., Thorup, K., Smith, J. A., & Swenson, G. W. (2007). Going wild: what a global small-animal tracking system could do for experimental biologists. *Journal of Experimental Biology, 210*(2), 181-186.

Wilson, A., Wikelski, M., Wilson, R. P., & Cooke, S. J. (2015). Utility of biological sensor tags in animal conservation. *Conservation Biology, 29*(4), 1065-1075.

Wilson, R. P., & McMahon, C. R. (2006). Measuring devices on wild animals: what constitutes acceptable practice? *Frontiers in Ecology and the Environment, 4*(3), 147-154.

Wu, Y., Lim, J., & Yang, M.-H. (2013). *Online object tracking: A benchmark.* Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.

Wu, Y., Lim, J., & Yang, M.-H. (2015). Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 37*(9), 1834-1848.

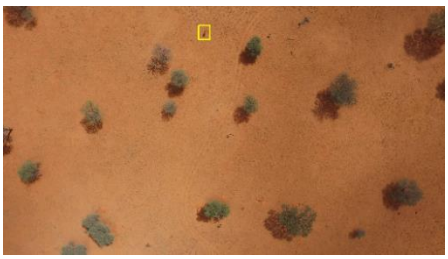
Yang, C., Duraiswami, R., & Davis, L. (2005). *Efficient mean-shift tracking via a new similarity measure.* Paper presented at the Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE computer society conference on.

Yilmaz, A., Javed, O., & Shah, M. (2006). Object tracking: A survey. *Acm computing surveys (CSUR), 38*(4), 13.

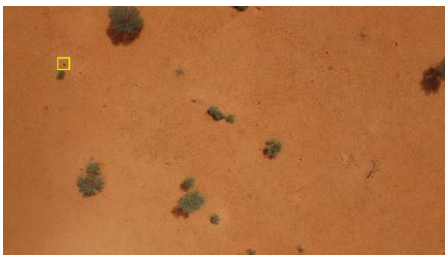
Zeppelzauer, M. (2013). Automated detection of elephants in wildlife video. *EURASIP journal on image and video processing, 2013*(1), 46.

Appendix I – Dataset description

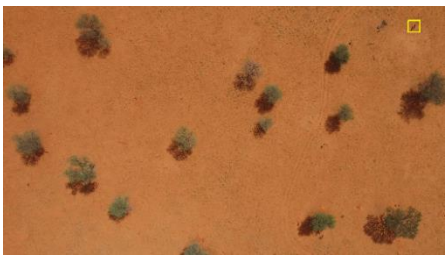
The images and tables give an overview of the objects and videos in the KWT dataset. The dataset consisted of 18 videos and 34 objects. The left image shows the overall scene, the middle image a close-up of one of the objects and the table provides some additional information. The colours in the table correspond with the colours used for the bounding boxes in the overall scene.




Video A1					
Features					
Animal #	1	2	3	4	5
Time (sec)	6				
N of frames	180				
Size (avg)	1392				
Velocity (avg)	10.73				

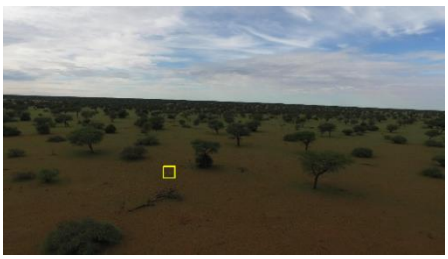



Video A2					
Features	SO				
Animal #	1	2	3	4	5
Time (sec)	9				
N of frames	264				
Size (avg)	905				
Velocity (avg)	5.16				

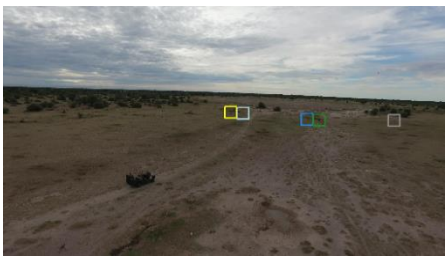



Video B1					
Features					
Animal #	1	2	3	4	5
Time (sec)	4				
N of frames	126				
Size (avg)	1096				
Velocity (avg)	15.71				

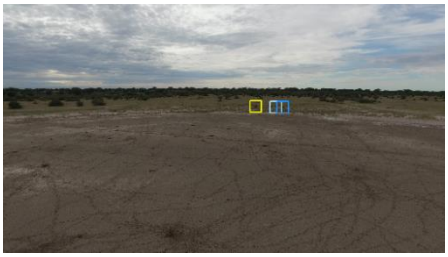

Video B2					
Features	COD				
Animal #	1	2	3	4	5
Time (sec)	2				
N of frames	49				
Size (avg)	1316				
Velocity (avg)	18.59				




Video C1					
Features	OCC - SO				
Animal #	1	2	3	4	5
Time (sec)	7				
N of frames	213				
Size (avg)	1105				
Velocity (avg)	12.52				



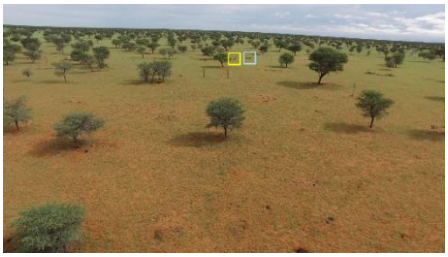

Video D1-D5					
Features	SO				
Animal #	1	2	3	4	5
Time (sec)	16	15	11	12	7
N of frames	455	440	312	347	197
Size (avg)	281	290	287	361	249
Velocity (avg)	4.16	4.08	3.93	3.30	2.36



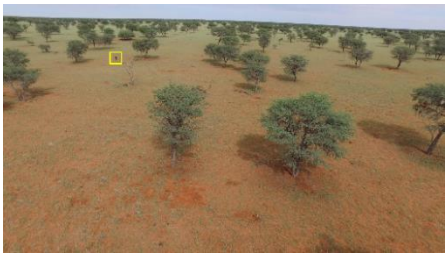
Video E1-E3					
Features	OCC - CM - SO				
Animal #	1	2	3	4	5
Time (sec)	11	29	31		
N of frames	322	844	898		
Size (avg)	4387	35621	11965		
Velocity (avg)	7.49	8.34	7.80		

Video F1					
Features	OR - CM - SO - COD				
Animal #	1	2	3	4	5
Time (sec)	53				
N of frames	1549				
Size (avg)	15617				
Velocity (avg)	10.45				

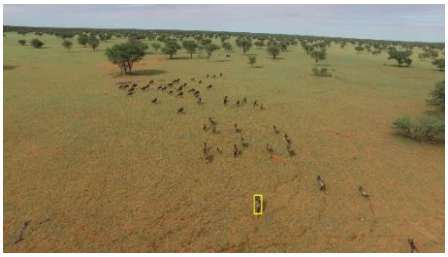
Video G1-G2					
Features	OCC - OR - CM - SO - COD				
Animal #	1	2	3	4	5
Time (sec)	45	44			
N of frames	1317	1281			
Size (avg)	2758	4044			
Velocity (avg)	10.00	8.57			



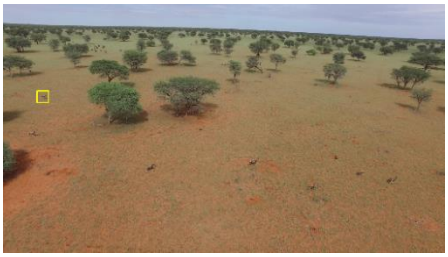
Video H1-H2					
Features	OCC - OR - SO - COD				
Animal #	1	2	3	4	5
Time (sec)	31	31			
N of frames	906	895			
Size (avg)	465	724			
Velocity (avg)	6.33	5.65			



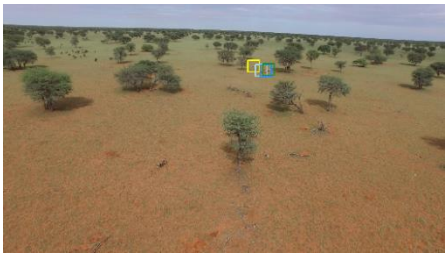
Video J1					
Features	OR - COD				
Animal #	1	2	3	4	5
Time (sec)	5				
N of frames	140				
Size (avg)	3255				
Velocity (avg)	9.58				



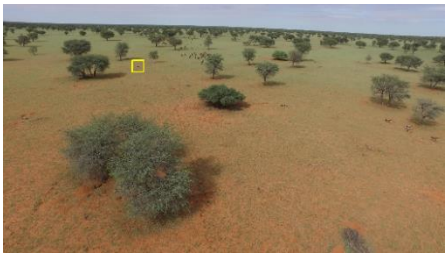
Video K1					
Features	OR - SO - COD				
Animal #	1	2	3	4	5
Time (sec)	22				
N of frames	643				
Size (avg)	18294				
Velocity (avg)	10.04				



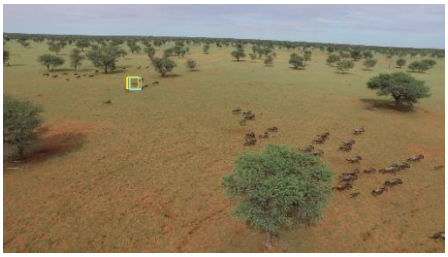
Video L1					
Features	OCC - SO				
Animal #	1	2	3	4	5
Time (sec)	8				
N of frames	228				
Size (avg)	6514				
Velocity (avg)	13.92				



Video M1-M4					
Features	SO				
Animal #	1	2	3	4	5
Time (sec)	3	3	3	3	
N of frames	96	78	80	82	
Size (avg)	2602	3833	2620	3021	
Velocity (avg)	17.74	20.88	19.75	19.00	

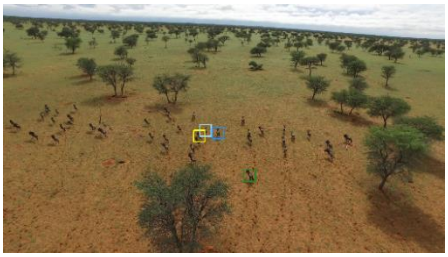


Video N1					
Features	OR - COD				
Animal #	1	2	3	4	5
Time (sec)	5				
N of frames	142				
Size (avg)	1638				
Velocity (avg)	11.12				



Video O1-O2					
Features	OCC - OR - SO - COD				
Animal #	1	2	3	4	5
Time (sec)	37	37			
N of frames	1086	1087			
Size (avg)	873	790			
Velocity (avg)	5.21	5.14			

Video P1-P2					
Features	OCC - OR - SO - COD				
Animal #	1	2	3	4	5
Time (sec)	54	54			
N of frames	1575	1558			
Size (avg)	2463	3397			
Velocity (avg)	4.28	4.38			



Video Q1-Q4					
Features	OCC - OR - SO - COD				
Animal #	1	2	3	4	5
Time (sec)	31	31	31	31	
N of frames	907	907	889	896	
Size (avg)	3947	5469	9269	13609	
Velocity (avg)	4.55	5.45	6.62	8.09	